Oracle Fusion Cloud Sales Automation

How can I modify standard Sales pages in Oracle Visual Builder Studio?

Oracle Fusion Cloud Sales Automation
How can I modify standard Sales pages in Oracle Visual Builder Studio?

F88585-08

Author: Jiri Weiss

Contents

Get Help

There are a number of ways to learn more about your product and interact with Oracle and other users.

Get Help in the Applications

Use help icons to access help in the application. If you don't see any help icons on your page, click your user image or name in the global header and select Show Help Icons.

Get Support

You can get support at *My Oracle Support*. For accessible support, visit *Oracle Accessibility Learning and Support*.

Get Training

Increase your knowledge of Oracle Cloud by taking courses at *Oracle University*.

Join Our Community

Use *Cloud Customer Connect* to get information from industry experts at Oracle and in the partner community. You can join forums to connect with other customers, post questions, suggest *ideas* for product enhancements, and watch events.

Learn About Accessibility

For information about Oracle's commitment to accessibility, visit the *Oracle Accessibility Program*. Videos included in this guide are provided as a media alternative for text-based topics also available in this guide.

Share Your Feedback

We welcome your feedback about Oracle Applications user assistance. If you need clarification, find an error, or just want to tell us what you found helpful, we'd like to hear from you.

You can email your feedback to *oracle_fusion_applications_help_ww_grp@oracle.com*.

Thanks for helping us improve our user assistance!

1 Before You Begin

This Guide Is for Customers on Update 24C

Customers who are still on Update 24B, should access the archived PDF version of the guide on the *Implement Next Gen Sales page on Oracle ScureSites*.

Release 24C expands the capabilities of CX Extension Generator. The now unnecessary manual steps have been removed from this version of the guide.

If you don't have access to SecureSites, you can ask an Oracle employee to provide it. See the *Getting Help section on Oracle SecureSites and SecureFiles*.

Before You Modify Pages in Visual Builder Studio

Before your team can start creating application extensions, you must first set up Visual Builder Studio and configure it. You only need to complete these steps once per implementation.

Before you begin, make sure you have the following information:

- The URL of your instance of Visual Builder Studio

  You can obtain the URL by navigating from your Oracle Cloud Applications development instance. From the Navigator, under **Configuration**, select **Visual Builder**.

Setting up Visual Builder Studio includes these steps:

1. Obtain the required roles and privileges.
2. Set up Visual Builder Studio users.
3. Create a project and workspace.

   Or, create a workspace in an existing project.
4. Create the build pipeline to deploy your changes to your production environment.

Get the Required Roles and Privileges

To set up Visual Builder Studio and your Oracle Cloud Applications instances, make sure you're assigned the FND_ADMINISTER_SANDBOX_PRIV privilege. Each role listed below, for example, has this privilege:

- Application Administrator (ORA_FND_APPLICATION_ADMINISTRATOR_JOB)
- Application Developer (ORA_FND_APPLICATION_DEVELOPER_JOB)
- Sales Administrator (ORA_ZBS_SALES_ADMINISTRATOR_JOB)
- Customer Relationship Management Application Administrator (ORA_ZCA_CUSTOMER_RELATIONSHIP_MANAGEMENT_APPLICATION_ADMINISTRATOR_JOB)

In addition, in order to view the **Edit Page in Visual Builder Studio** option in the Settings and Actions menu, users must be assigned the View Administration Link privilege (FND_VIEW_ADMIN_LINK_PRIV). By default, this privilege is assigned to the predefined roles listed above. Grant this privilege to custom roles, as well, where required.

Set Up Visual Builder Studio Users in IDCS

First, you must authorize users so they can access Visual Builder Studio to do extension work. You do this in Oracle Identity Cloud Service (IDCS).

To sign in to the right instance of IDCS, complete these steps:

1. Navigate to www.oracle.com.
2. Click **View Accounts**.
3. Click **Sign in to Cloud**.
4. In the Cloud Account Name field, enter the name of your production pod, such as **cakp**. Click **Next**.
5. The Oracle Cloud Account Sign In page is specific to the Cloud Portal and is separate from your test and production pods. Enter your user credentials and click **Sign In**, or use the **Click here** link to reset your password.

   On the Oracle Cloud Dashboard, navigate to the Active Applications region and click **Identity Cloud**. Scroll down to view your IDCS service instances:

   - Identity

     IDCS instance associated with the Myservices portal.
   - CAKP (for example)

     IDCS instance associated with your production pod.
   - CAKPTEST (for example)

     IDCS instance associated with your developer pod.
6. To sign in to the IDCS instance associated with your test environment, click the Service Instance URL for your test pod (in this example, CAKPTEST), or click the **Open Service Console** link.

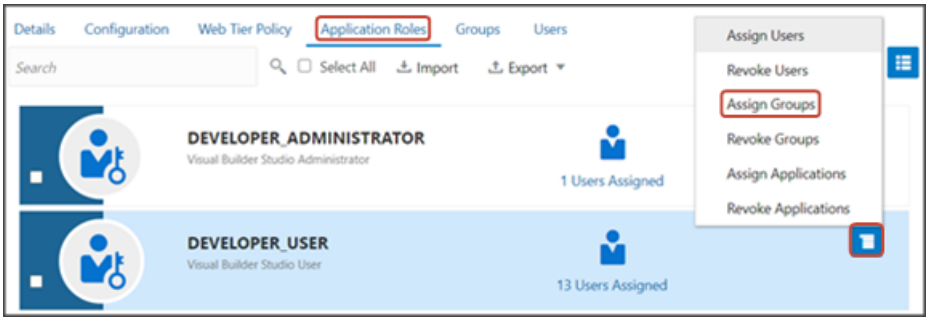
   > **Tip:** Bookmark this URL so you have it for next time.
7. Sign in to IDCS directly, or click **Oracle Applications Cloud (Fusion)** at the bottom of the page to sign in using your Oracle Applications user credentials.

Once you're in the Identity Cloud Service console, you can now authorize users so they can access Visual Builder Studio. You can assign users, or groups of users, to the Visual Builder Studio administrator (DEVELOPER_ADMINISTRATOR) or Visual Builder Studio user (DEVELOPER_USER) roles. These two predefined roles are specific to working with Visual Builder Studio.

In this example, let's look at assigning a group to the Visual Builder Studio administrator (DEVELOPER_ADMINISTRATOR) role. This is a way to provide Visual Builder Studio access to Fusion Application roles and their users.

1. In the Identity Cloud Service console, expand the Navigation Drawer, and then click **Oracle Cloud Services**.
2. Click **DevServiceAppAUTO_<VBSTUDIO_INSTANCE>**.
3. Click the Application Roles tab.

4. On either the DEVELOPER_ADMINISTRATOR or DEVELOPER_USER row, click the menu icon, then click **Assign Groups**.



5. In the Assign Groups dialog, select the roles that you want to assign and click **OK**. For example:
    - Application Administrator
    - Sales Administrator
    - Customer Relationship Management Application Administrator
    - Application Developer
    - Any custom role, per your authorization plan

Users with the selected roles can now access Visual Builder Studio. Note that sometimes it may take up to 12 hours for Oracle Cloud Applications user updates to sync with IDCS.

For more information, see *Set Up VB Studio Users* in the Oracle Cloud Administering Visual Builder Studio guide.

Create Your Project and Workspace

If you haven't yet created a project and workpace in Visual Builder Studio, then you must do so before configuring a standard object's detail page.

- A **project** is a collection of resources that your team will use to create the application extension.

  > **Note:** The procedures in this chapter use fragments which are building blocks that help you to configure pages more quickly, with a minimal amount of manual coding. Create a new project to keep your fragment-based extension separate from extensions that don't use fragments.

- A **workspace** exists within a project, and is your private work area which connects to a Git repository, environment, and Oracle Cloud Applications sandbox.

If you **already** created a project, then you don't need to create a new one. Rather, you can create a new workspace within your existing project. See the next section.

To create a project and workspace at the same time:

1. In Visual Builder Studio, click the Organization side tab, then click the Projects subtab.
2. Click **+ Create**.
3. In the New Project dialog, enter a name and description for the project.
4. Set the security for your project, either **Private** or **Shared**.
5. Click **Next**.
6. Select **Empty Project** as the template for this project, and click **Next**.
7. On the Properties step, click **Next**.

8. On the Team step, add team members and select their membership levels.
9. Click **Finish**.

   Visual Builder Studio provisions your project, which could take a few minutes to complete. You will then be navigated to the project's home page.
10. Create a development environment for your project.
    a. Click the Environments side tab, then click **+ Create Environment**.
    b. Enter a name and description for the environment and click **Create**.
    c. Click **+ Add Instance**.
    d. In the Add Service Instances dialog, under Instance Type, click **Oracle Cloud Applications**.
    e. Under Authentication Method, click **Identity Domain**.
    f. Select the desired instance for the environment and click **Add**.
11. Create a new workspace.
    a. Click the Workspaces side tab, then click **New** > **New Application Extension**.
    b. In the New Application Extension dialog, in the Extension Name field, enter the extension name.
    c. In the Extension Id field, enter `site_cxsales_Extension`.
    d. In the Workspace Name field, accept the default value or enter a new workspace name.
    e. In the Development Environment field, select the environment you just created.
    f. In the Base Oracle Cloud Application field, select **CX Sales (from CX Sales)**.
    g. In the Git Repository region, click **Create new repository**.
    h. Enter a name for both the repository and working branch.
    i. Click **Create**.

Create a Workspace in an Existing Project

To create a new workspace in an existing project:

1. In Visual Builder Studio, click the Workspaces tab on the left pane, then click **Clone from Git**.
2. In the Clone from Git dialog, enter these values:

Field	Value
Workspace name	Enter the name of your workspace.
Repository Name	Select the Git repository connected to your team's project. Ask your administrator for this value.
Branch	Select **main**.
"New branch from selected" check box	Select this check box.
New branch name	Enter a name for your branch. For example, **<user_id>_<feature_name>**.
Development Environment	Select your development environment.

Field	Value
Sandbox (optional)	Select a sandbox, or leave this blank and attach a sandbox to your workspace later.

3. Click **Create**.

Create the Build Pipeline for Your Production Environment

Create the build pipeline to deploy your changes to your production environment.

When you created your project, a build pipeline was automatically created to deploy your application extension to your test environment. Now you must create a second pipeline for your production environment.

For complete instructions, see *Configure VB Studio to Run CI/CD Pipelines* and *Set Up the Project to Deploy to Production* in the Oracle Cloud Administering Visual Builder Studio guide.

Related Topics

- Setting Up Visual Builder Studio for Oracle Cloud Application Extensions

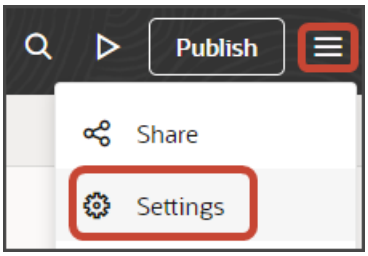
How can I change my project's Extension ID?

When working with a Sales application extension in Oracle Visual Builder Studio, your project's extension ID must be `site_cxsales_Extension`. This topic illustrates how to correct the extension ID, if required.

To change the extension ID for a project:

1. In Visual Builder Studio, from the left navigator, click **Environments** > **Deployments**.
2. Undeploy any deployments to target servers.

   See *Delete an Extension*.
3. Navigate to your workspace and, from the upper menu, click **Settings**.



4. In the **Extension ID** field, enter `site_cxsales_Extension`.
5. Build and deploy your extension once more.

   See *Build Your Applications* and *Package, Deploy, and Manage Extensions*.

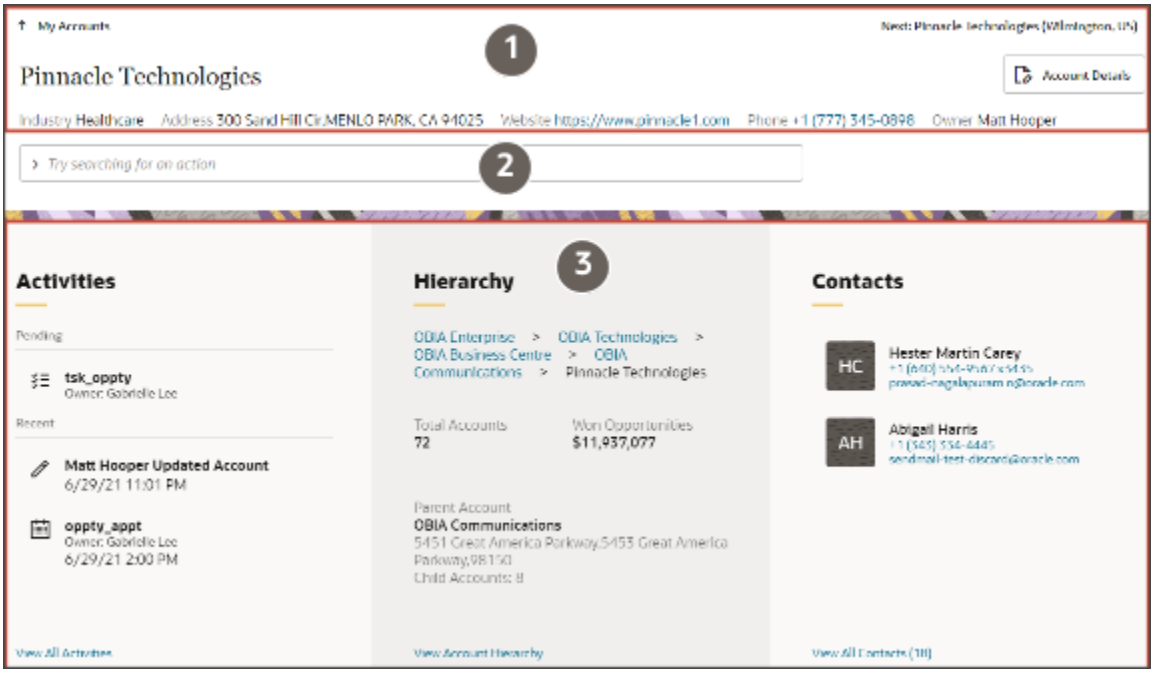
2 Configure Standard Object Pages

What You Can Configure

An individual detail page is divided into three sections: the header, the Action Bar, and a panels region (also known as the foldout). You can use Oracle Visual Builder Studio to configure a standard object's header and panels region so that the page shows the exact information that your users need. You can use CX Extension Generator in Application Composer to simplify the process of adding panels and subviews to standard objects. Using another tool, called Dispatcher, you can change which page opens when users click name links in the details and list pages.

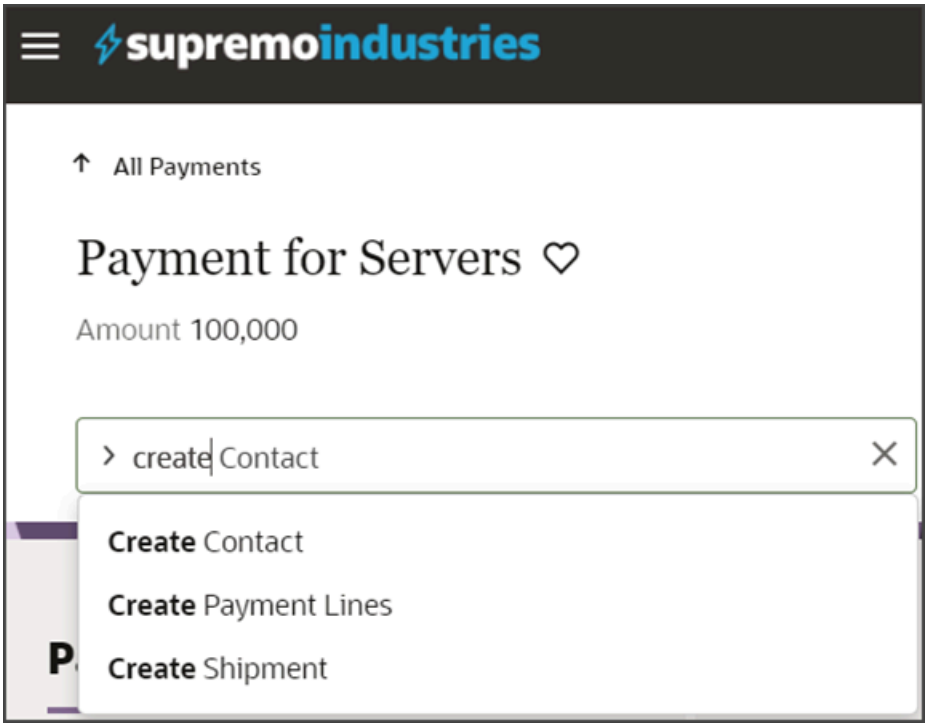
Note: In the runtime UI the detail page is called the overview page.

Here's a screenshot of the three main regions on a detail page. You can configure both the header and the panels region (callouts 1 and 3 in the screenshot). The topics in this chapter illustrate the various configuration options available.



Smart Actions in the Action Bar

The Action Bar is a field at the top of many pages where users can type keywords to access and update information, and take actions. For example, here's a screenshot of the Action Bar with suggested actions, after typing `create`:



Standard objects are delivered with a set of preconfigured smart actions, but you can also create your own using Application Composer.

> **Note:** If you previously created custom smart actions for a non-fragments implementation, then you don't need to create new smart actions for use with fragments. Instead, update existing UI-based custom smart actions to specify the action type, either **Add** or **Create**, as well as the target object and any required field mapping. For existing REST-based or object function-based smart actions that you already created, edit the action and then save without making any changes. These steps ensure that your custom smart actions still work with new fragment-based extensions.

Prerequisites

Before configuring a standard object's detail page, make sure you have a project and workspace ready in Visual Builder Studio.

For instructions about how to create a project and workspace, see the Before You Begin chapter.

Configure the Header on the Detail Page

The header on a record's detail page displays key details right at the top, so that users can quickly find what they need. The header displays a predefined set of attributes, but you can configure the region to display any attributes that your users are most interested in.

Where's the Header?

When you drill down to a record's detail page from a list page, the header displays at the top of the detail page.

Configure the Header Region Using a Rule Set

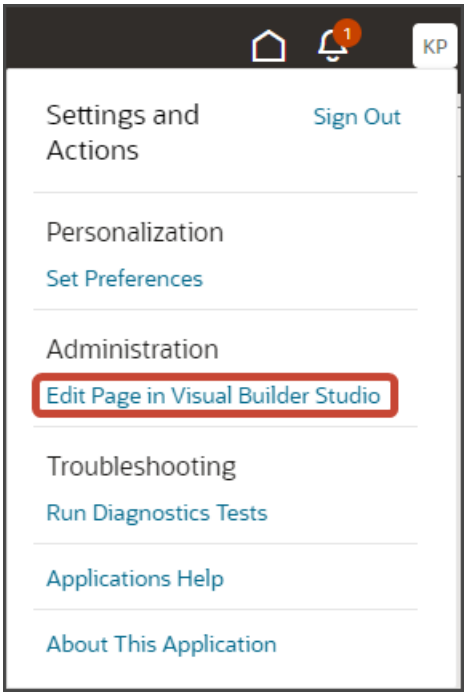
You can modify the header using Oracle Visual Builder Studio. To modify any region on a page, you create a new rule that returns a custom layout.

In this example, let's look at adding the Type field to the header on the account detail page.

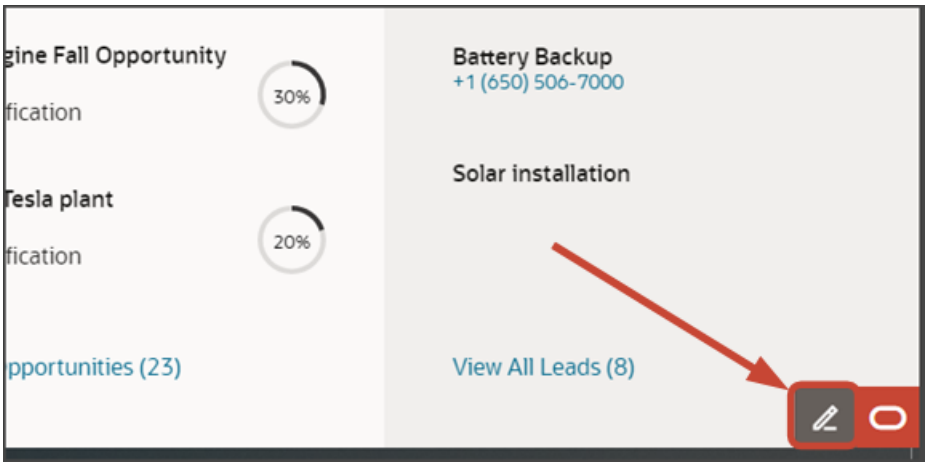
1. Navigate to the page that displays the area you want to extend.

   In this example, navigate to the My Team's Accounts page and then click an account record.

2. To access Visual Builder Studio, you have one of two options:
    - Under the Settings and Actions menu, select **Edit Page in Visual Builder Studio**.



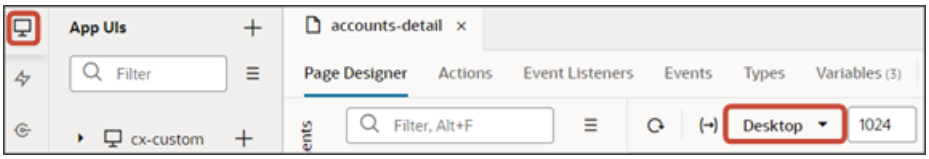
    - If you don't see the Settings and Actions menu, then in the lower right corner of the page, hover over the Ask Oracle icon and click **Edit Page in Visual Builder Studio**.



3. If you're working on multiple projects, then select the Extend Redwood CX Sales project to keep your fragments extension separate from non-fragments changes.
4. If you're working in an active sandbox when you launch Visual Builder Studio, then Visual Builder Studio looks for a workspace that's associated with your sandbox. If you're not working in a sandbox, then Visual Builder Studio looks for a workspace without a sandbox. You might have to select a workspace if more than one workspace exists. If no workspace exists, then Visual Builder Studio automatically creates one for you.

5. When you enter into your workspace in Visual Builder Studio, the accounts-detail page is displayed in the Page Designer canvas.

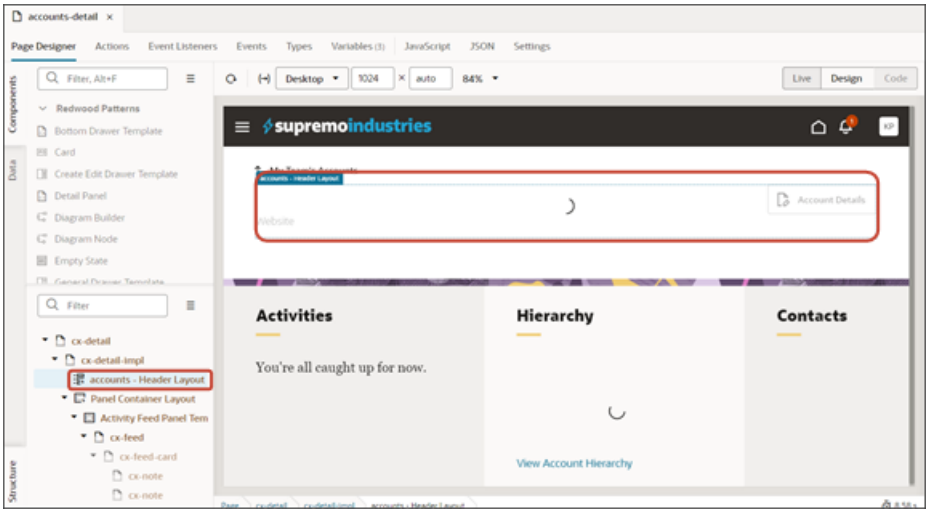
> **Tip:** To display more of the accounts-detail page in the canvas, collapse the left pane by clicking the App UIs tab on the left. Then, select **Desktop** or **Wide screen** from the list at the top of the Page Designer.



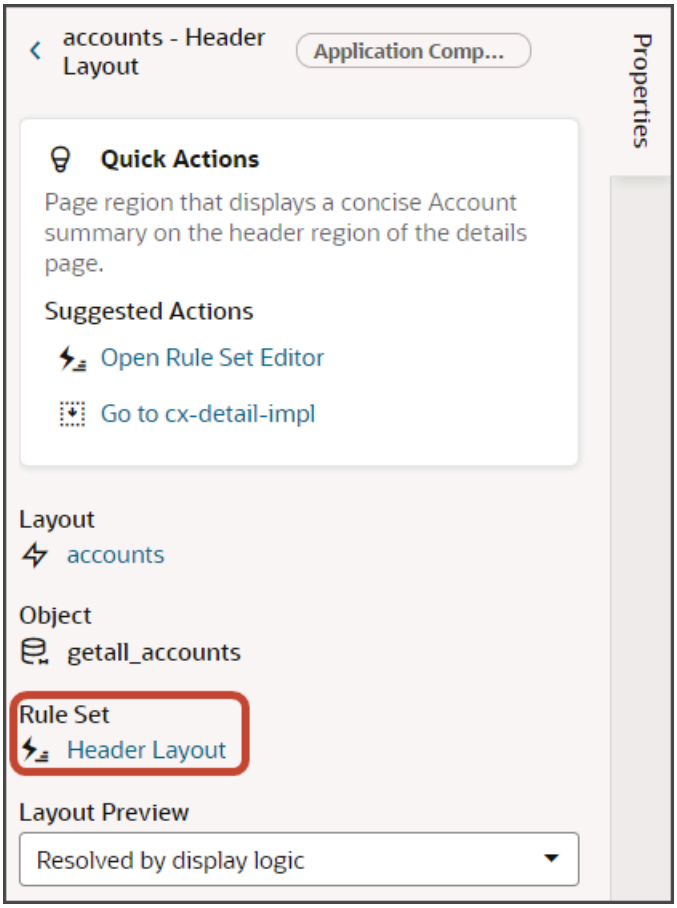
6. Click the **Design** button.

7. On the Structure panel, click the **accounts - Header Layout** node.

   The header is highlighted in the canvas.

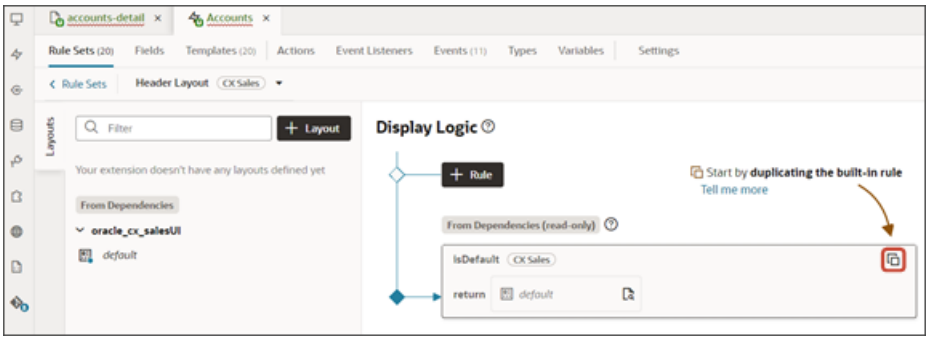


   On the Properties pane, the rule set is also displayed.



8. On the Properties pane, click the **Header Layout** rule set.

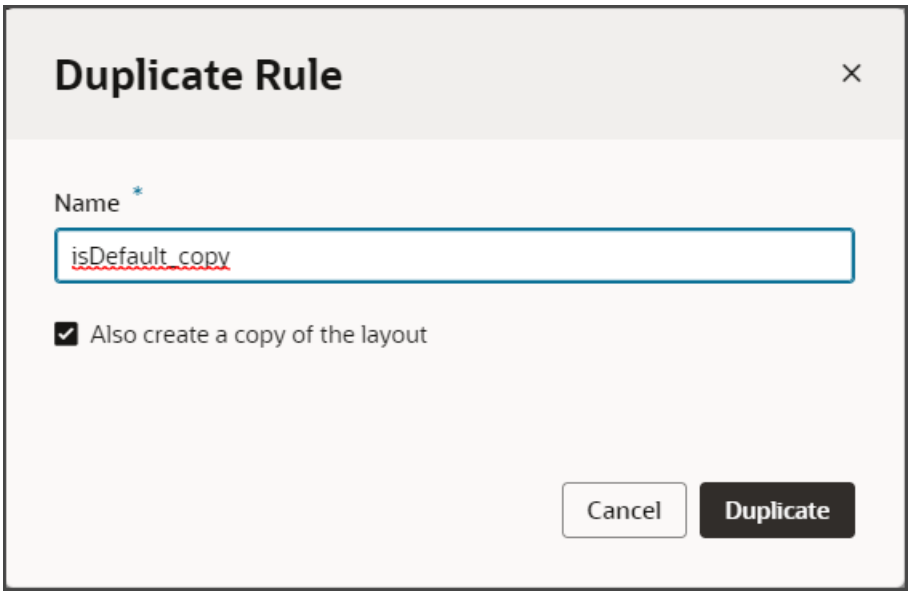
   The Accounts layout tab > Rule Sets subtab displays. The Rule Sets subtab includes both a default layout as well as a default rule. Let's create a new rule.



9. Click the Duplicate Rule icon.

10. In the Duplicate Rule dialog, accept the default rule name or enter a new name. The name you enter here is both the rule name and also the layout name, so enter a layout name that makes sense for you.

    Also, make sure that the **Also create a copy of the layout** check box is selected.

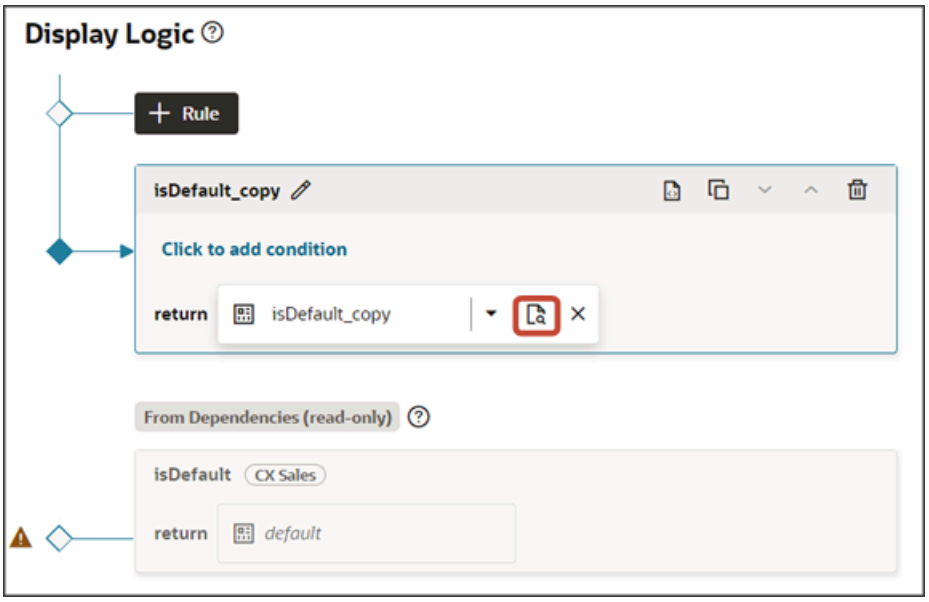


11. Click **Duplicate**.

The new rule displays at the top of the list of existing rules, which means that this rule will be evaluated first at runtime. If the rule's conditions are met, then the associated layout is displayed to the user.

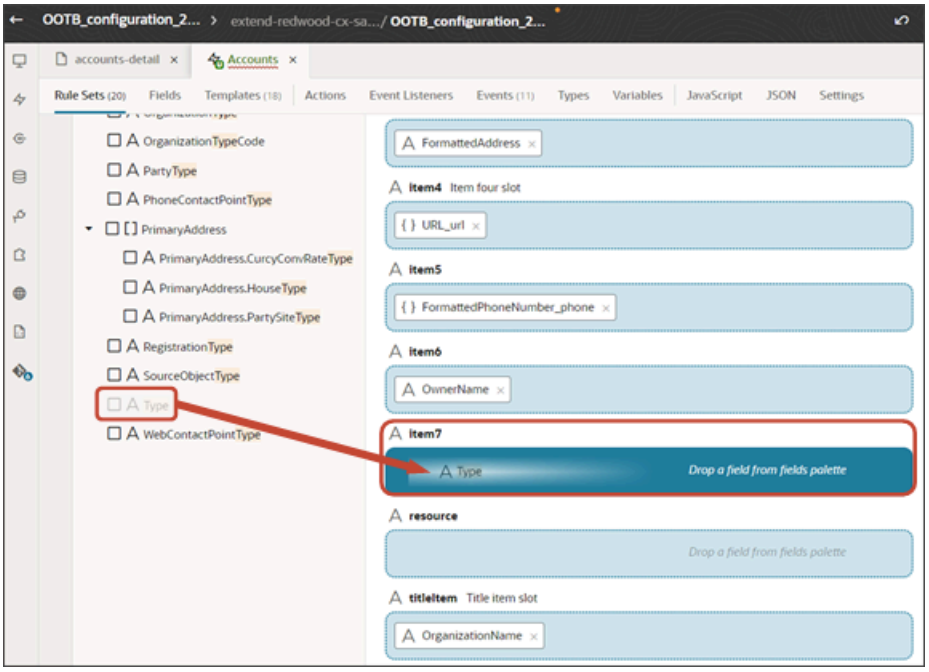
In this example, we're not adding any conditions which means that the associated layout will always be displayed.

12. Modify the rule's copied layout.
    a. Click the Open icon to edit the copied layout.



    b. In the Filter field, enter `type` to locate the Type field.
    c. Drag the Type field to an available section in the header.

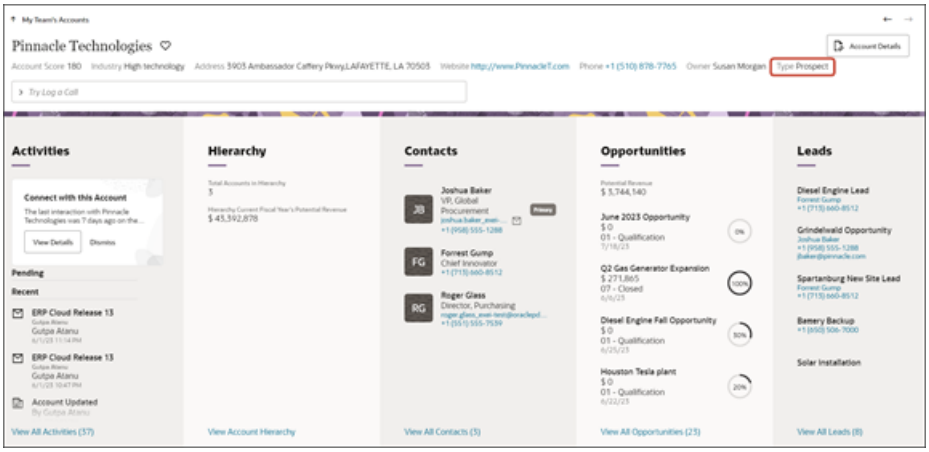
    In our example, drag the Type field to the item7 section.



13. Click the Preview button to see your changes in your runtime test environment.

The preview link must include the `application/container` segments in the URL. If not, then change the preview link using the following example URL:

```
https://<servername>/fscmUI/redwood/cx-sales/application/container/accounts/accounts-detail?id=300000003513233&puid=7050&view=foldout
```



> **Note:** The header displays a field only if the field has a value. For example, if an account doesn't have a Type specified, then the Type field won't display.

Show or Hide a Field Based on Another Field's Value

In Oracle Visual Builder Studio, you can use rules and display logic to determine which page layout to display at runtime. You can also set up your rules so that a field only displays depending on the value of another field. For example, maybe you want the Owner field to always display for an account record, unless the account type is Prospect. This topic illustrates how to implement this scenario using display logic.

This topic also provides an alternative method for simple hide/show scenarios.

Prerequisite

In this example, we'll show or hide a field based on an account's type. Before you try the following procedure, you must add the Type field to the account detail page's header using the instructions documented in *Configure the Header on the Detail Page*.

Add Display Logic to a Rule

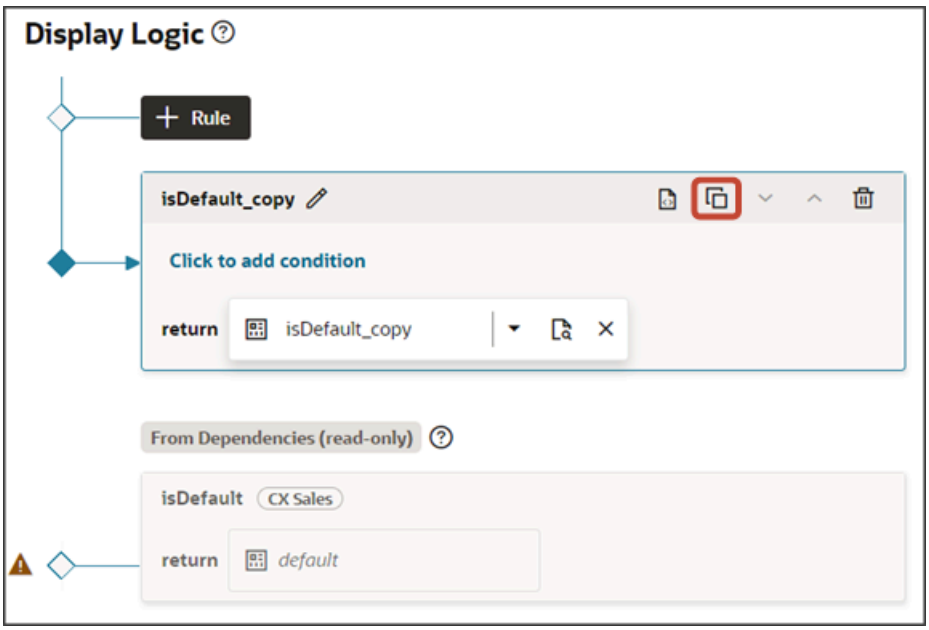
To show a field depending on the value of another field, we'll create two rules and add a condition to the first rule, as follows:

1. If an account's type is Prospect, then display a layout without the Owner field.
2. If an account's Type is not Prospect, then display a different layout that includes the Owner field.

Let's create the rule and corresponding layout for item #1.

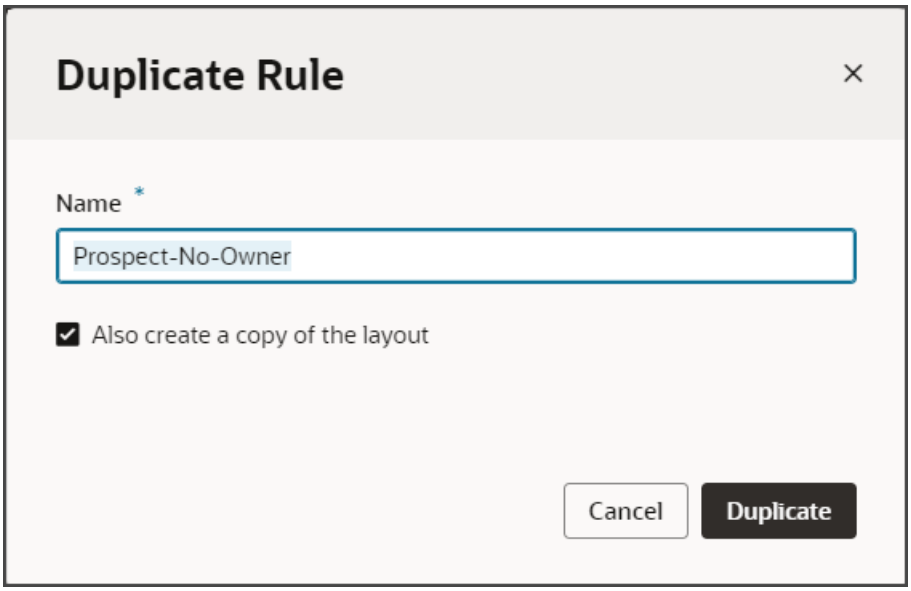
1. In Visual Builder Studio, on the Accounts layout tab > Rule Sets subtab, create a new rule by copying the rule that you created in the procedure documented in *Configure the Header on the Detail Page*.

   Click the Duplicate icon.



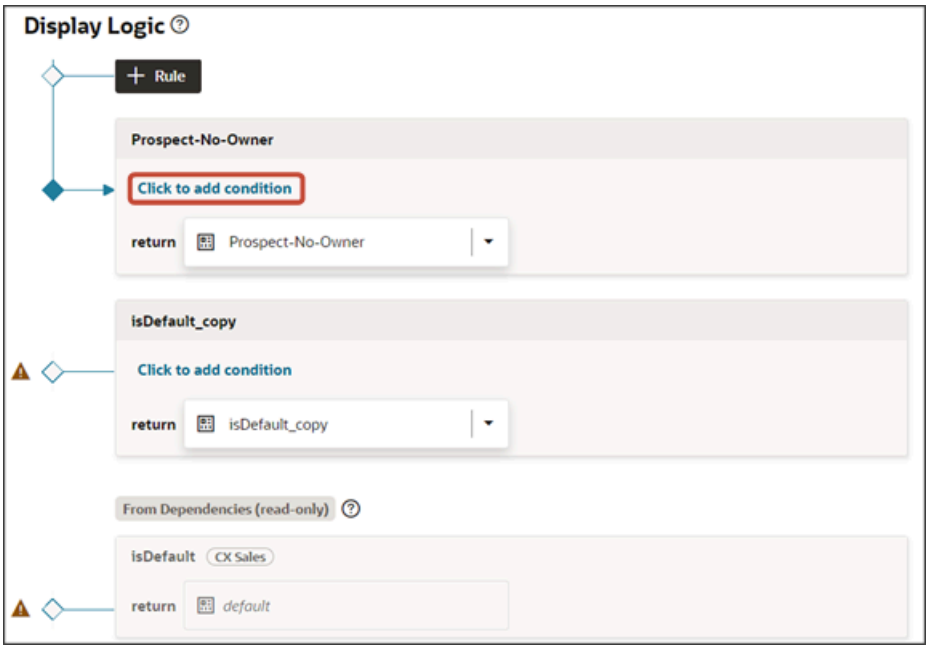
2. In the Duplicate Rule dialog, accept the default rule name or enter a new name. The name you enter here is both the rule name and also the layout name, so enter a layout name that makes sense for you, such as `Prospect-No-Owner`.

   Also, make sure the **Also create a copy of the layout** check box is selected.



3. Click **Duplicate**.

   The new rule displays at the top of the list of existing rules, which means that this rule will be evaluated first at runtime. If the rule's conditions are met, then the associated layout is displayed to the user. Otherwise, the next rule will be evaluated.
4. Let's add a condition to the rule. Click **Click to add condition**.

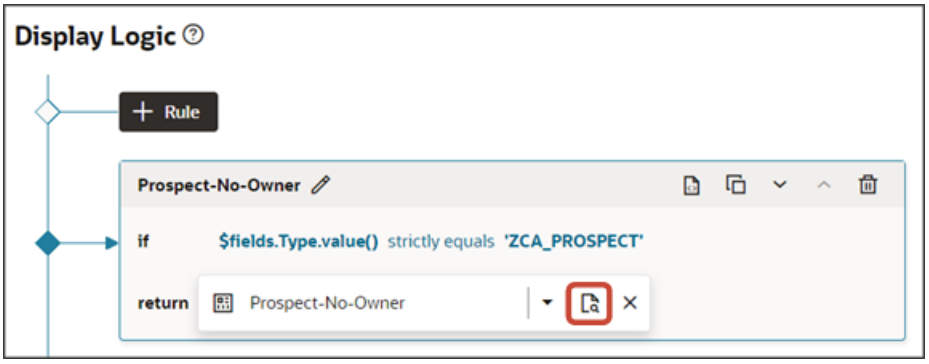


5. Add the condition to specify when you want the associated layout to display at runtime.

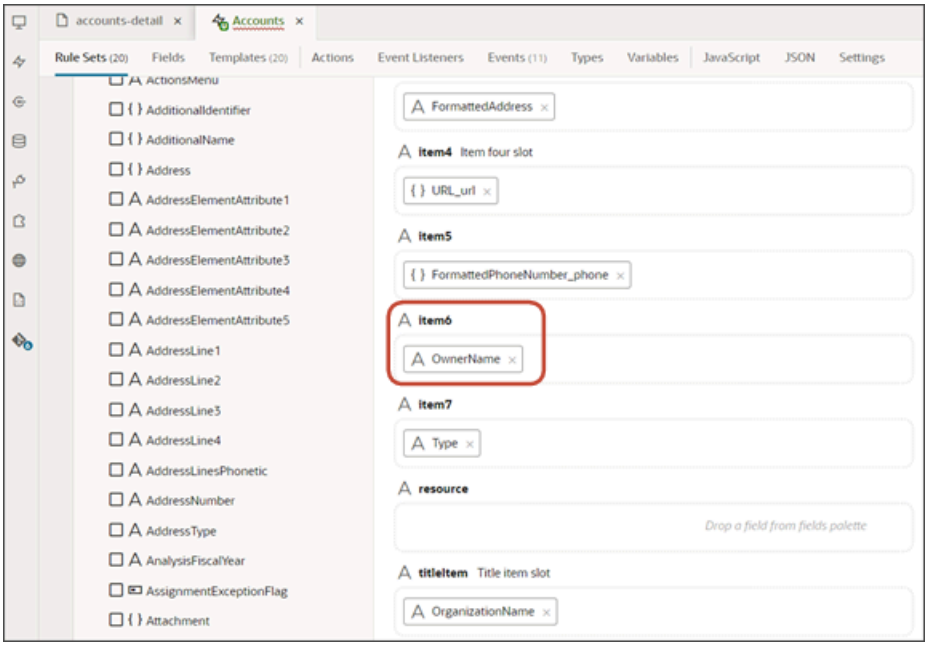
   In this example, we want the Type field to equal `Prospect`.

6. Edit the layout so that the Owner field is not included when the account type is Prospect.

   a. Click the Open icon to edit the copied layout.



   b. Delete the Owner field in the item6 slot.



7. Click the Preview button to see your changes in your runtime test environment.

The preview link must include the `application/container` segments in the URL. If not, then change the preview link using the following example URL:

```
https://<servername>/fscmUI/redwood/cx-sales/application/container/accounts/accounts-detail?id=300000003513233&puid=7050&view=foldout
```

In this example, we can see that this account is a prospect, and therefore there is no Owner field.

However, this account is a Customer, and thus we see the Owner field.

> **Note:** The header displays a field only if the field has a value. For example, if an account doesn't have a Type specified, then the Type field won't display.

> **Note:** The rule and corresponding layout that you previously created in *Configure the Header on the Detail Page* satisfy the conditions for item #2, "If an account's Type is not Prospect, then display a different layout that includes the Owner field."

Alternative Method for Simple Hide/Show Use Cases

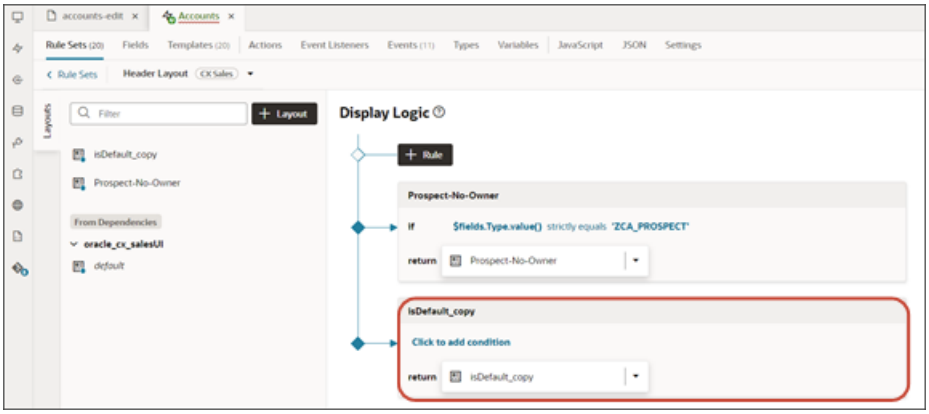
The previous example illustrates how to use rules, display logic, and custom layouts to show or hide a field based on another field's value. But, you can also directly modify the page template instead. This method is an alternative method that might be preferable for simple hide/show use cases.

Let's look at an example. In this next procedure, we'll hide the account's phone number if the account's industry is High Tech.

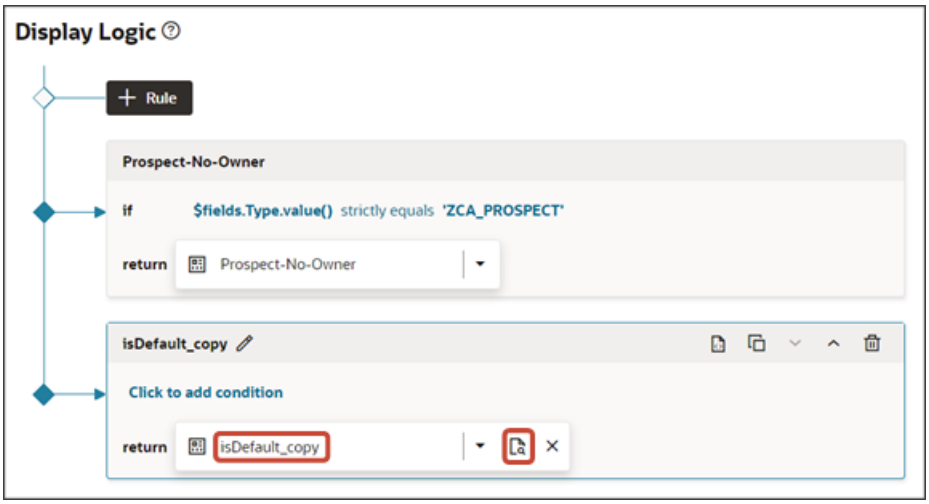
1. In Visual Builder Studio, on the Accounts layout tab > Rule Sets subtab, review the list of rules.

   We can ignore the Prospect-No-Owner rule because that rule is specifically for the instance when the Type field is Prospect.

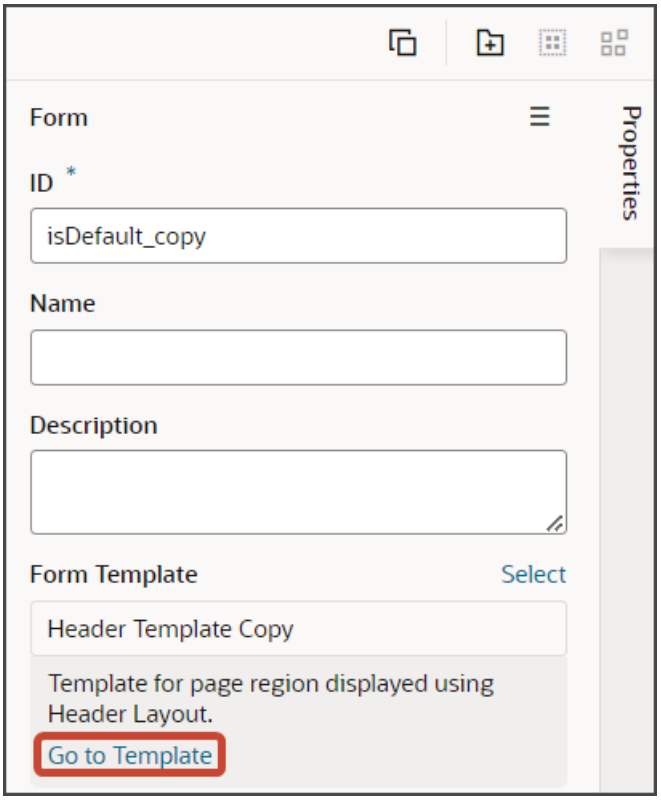
   Instead, we'll edit the rule that you created in the procedure documented in *Configure the Header on the Detail Page*.



2. Click the Open icon to edit the rule's copied layout.

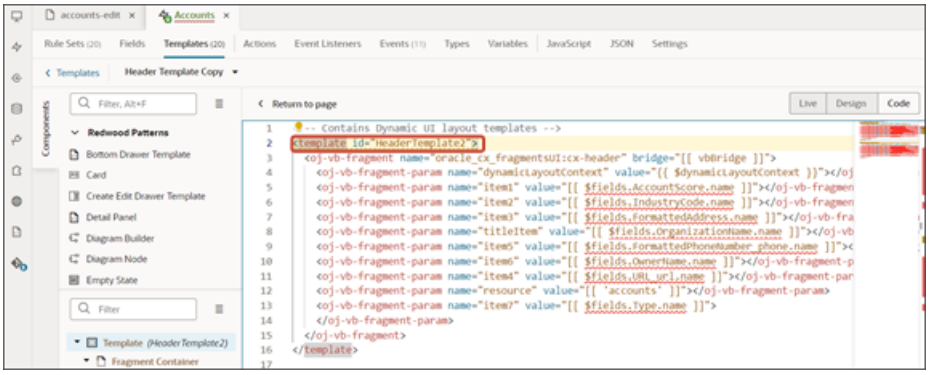


3. On the Properties pane, click **Go to Template**.

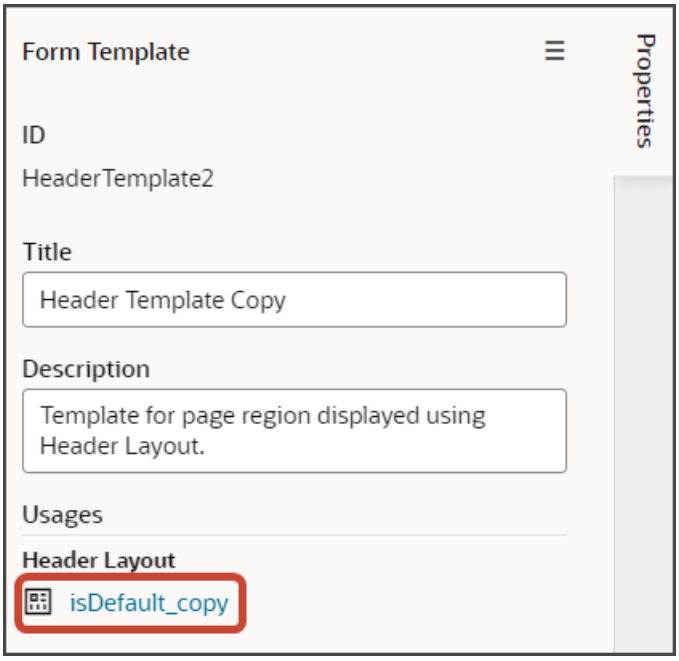


4. Click the Code button.

5. In the template editor, highlight the template tags for the header template.



6. On the Properties pane, confirm the layout name.



7. In the template editor, highlight the **FormattedPhoneNumber_phone** parameter.

```
<oj-vb-fragment-param name="item5" value="[[ $fields.FormattedPhoneNumber_phone.name ]]"></oj-vb-fragment-param>
```

8. Replace the highlighted parameter with this line:

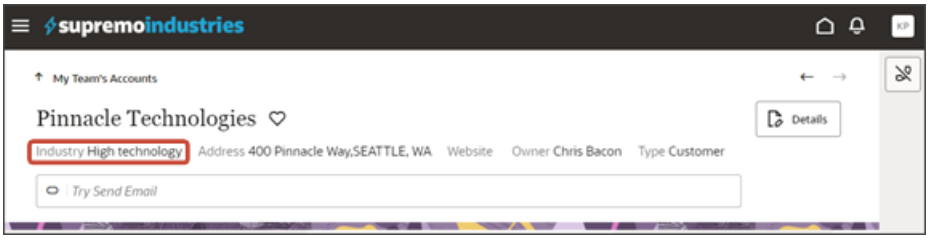
```
<oj-vb-fragment-param name="item5" value="[[ ( $fields.IndustryCode.value() =='HIGH_TECHNOLOGY') ? null : $fields.FormattedPhoneNumber_phone.name]]"></oj-vb-fragment-param>
```

9. From the accounts-detail page, click the Preview button to see your changes in your runtime test environment.

The preview link must include the `application/container` segments in the URL. If not, then change the preview link using the following example URL:

```
https://<servername>/fscmUI/redwood/cx-sales/application/container/accounts/accounts-detail?id=300000003513233&puid=7050&view=foldout
```

Let's look at an account. In this example, the account's industry is **High technology**, so the phone number doesn't display in the header.



If we change the industry to **Automotive**, notice that the phone number now displays.

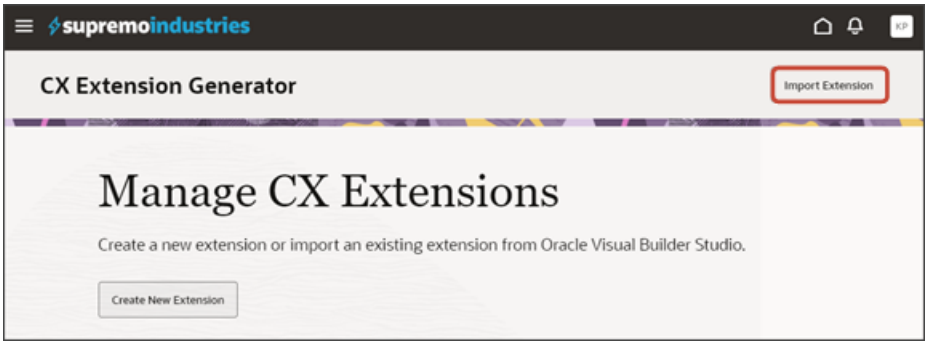
> **Note:** The header displays a field only if the field has a value. For example, if an account doesn't have a Type specified, then the Type field won't display.

Add Panels and Subviews to Standard Sales Objects

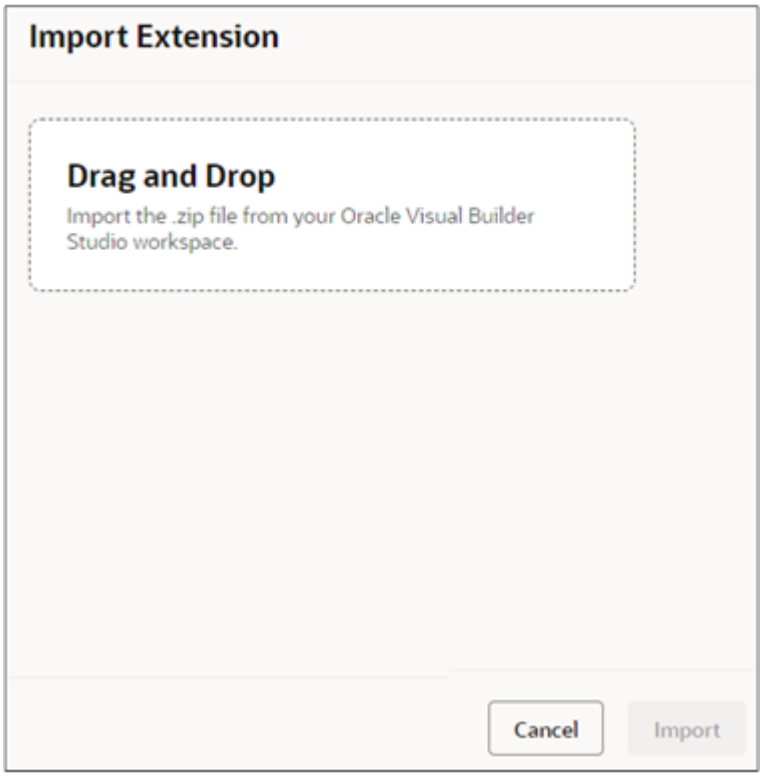
You can add panels and their subviews to the detail pages of standard objects such as accounts, opportunities and contacts. You can add both custom and standard objects as child objects. You can use CX Extension Generator to generate the UIs and smart actions and then add these to a custom layout for the object in Visual Builder Studio (VBS).

Create Panels, Subviews, and Smart Actions in CX Extension Generator

1. Create and enter a sandbox with Application Composer as a tool.
2. Open **Application Composer**.
3. Click **CX Extension Generator**.
4. If you've already configured your application in Visual Builder Studio for this environment, then you must import the configuration you exported from Visual Builder Studio.
    a. Click **Import Extension**.



b. In the Import Extension drawer, drag and drop your .zip file and click **Import**.



The existing objects in your application are now visible in the CX Extension Generator.

5. If this is the first time you're configuring the application, then click **Create New Extension**.

> **CAUTION:** You can use the **Create New Extension** button only the first time you configure your application in the environment. If you use the CX Extension Generator to make further changes after your initial upload to VBS, then you must import the files back from VBS using the **Import Extension** button before you start. If you create a new extension using the **Create New Extension** button and import your changes to VBS, then your upload will overwrite all your previous changes.

6. On the **Add Objects** drawer, select the object you're adding panels and subviews to.
7. Click **Add**.

8. On the CX Extension Generator page, click the object name link to add panels and subviews.

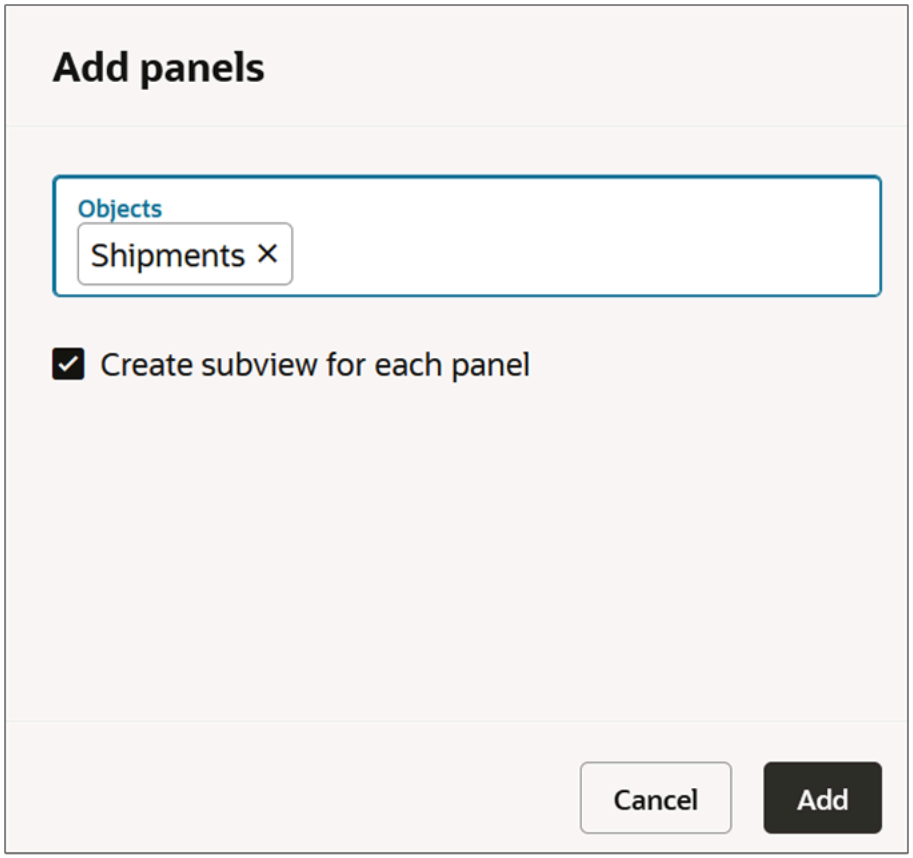
> **Note:** The Extension Generator doesn't display any of the existing panels. It displays only the panels you add.

9. To add a panel:

   a. On the default Configure Panel, click **Add**.

   b. In the Add Panels drawer, select the objects that you want to create the panels and subviews for. You can select standard or custom objects with 1:M, M:M, or child relationships.

      For example, if you're adding a panel to Account for the custom object Shipment, then add **Shipments**.

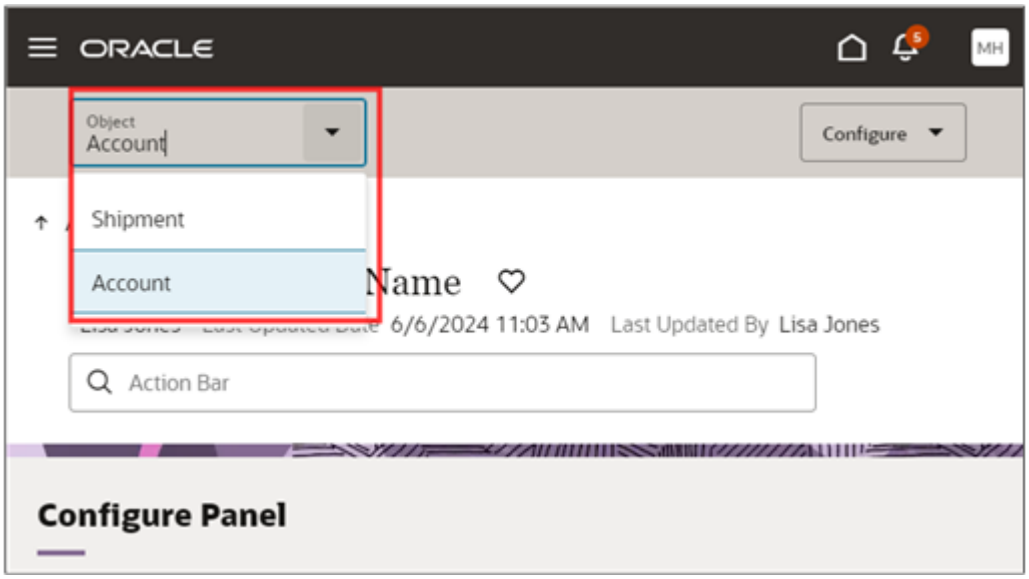


   c. Select the **Create subview for each panel** checkbox to automatically create a subview along with each panel.

> **Note:** If you don't select this checkbox, then you can add subviews later.

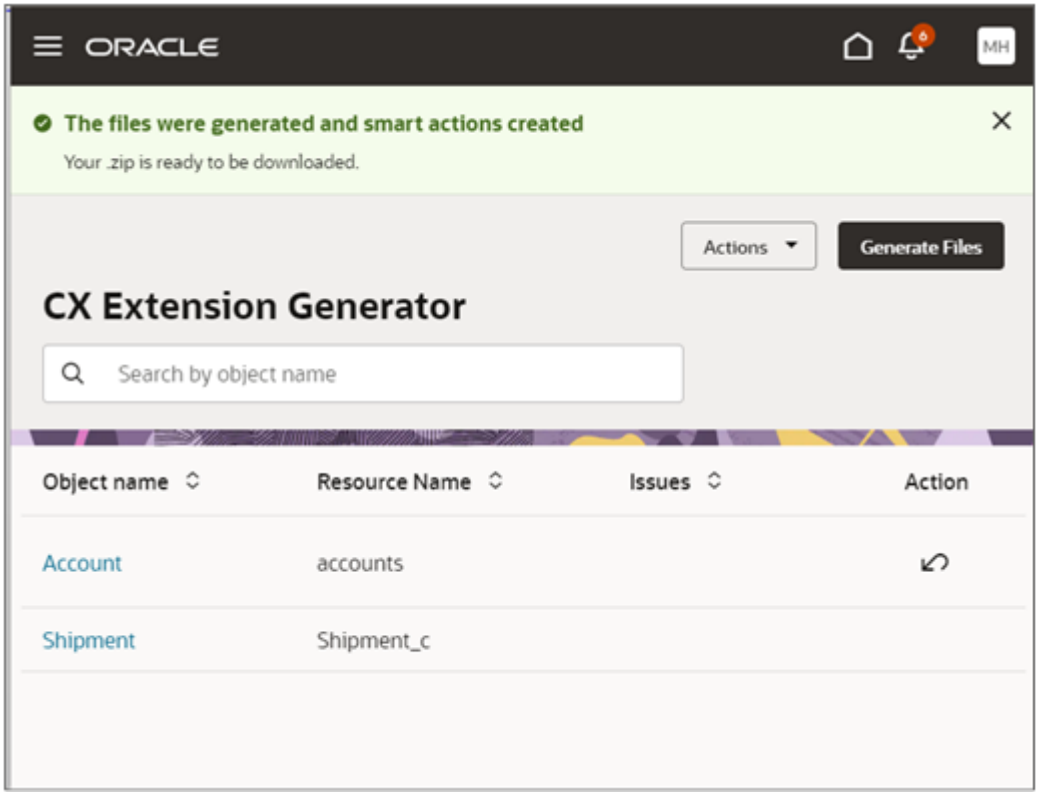
d. Click **Add**.

10. Optionally, click **Configure** and select **Smart Actions** or **Subviews** to review the smart actions and subviews that the Extension Generator will automatically create for the objects that you selected.
11. If you're adding a panel for a custom object and you haven't yet configured the panels and subview for it, then switch to that object by making a selection in the **Object** drop-down field.



12. After you've completed adding the panels, click the **All Objects** link.
13. Click **Generate Files**. The process can take a couple of minutes to complete.

14. When you see the **The files were generated and smart actions created** message, then your file is already downloaded. Check your downloads directory for the **Custom.zip** file.



Import the .Zip File into Visual Builder Studio

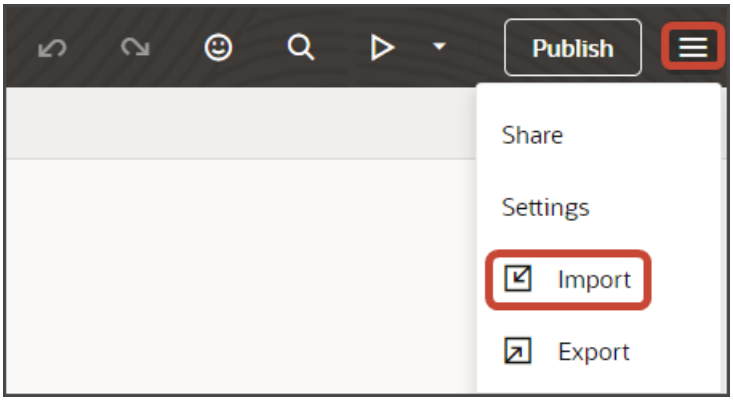
Import the .zip file created by the CX Extension Generator into Visual Builder Studio.

1. Use the Navigator to navigate to Visual Builder Studio: **Configuration** > **Visual Builder**.
2. In Visual Builder Studio, create a new workspace if you haven't already.

> **Note:** The workspace **Extension ID** must be **site_cxsales_Extension**. For information on how to change the extension, see *How can I change my project's Extension ID?*.

To create a workspace, see *Before You Create an Application Extension*. Be sure to follow the instructions for updating the extension ID for your workspace.

3. Click the Menu icon at the of the page, then click **Import**.



4. In the Import Resources dialog, add your .zip file and click **Import**.

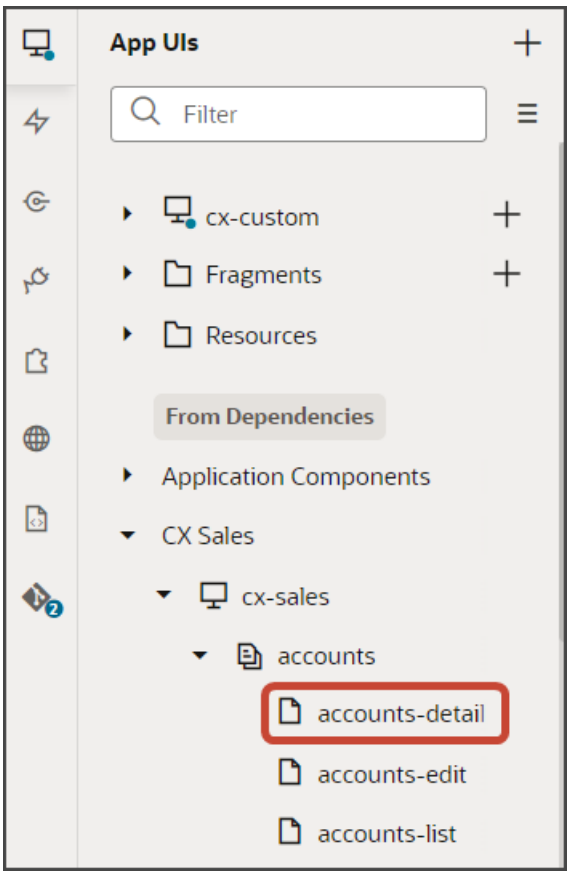
   Leave the **Delete existing files and resources** option unselected.

Create a Custom Panel Container for the Standard Object and Add the Imported Panels

Because you're adding panels and their subviews to an object provided by Oracle, you can't edit the object directly. You must create a copy and add configure the copy.

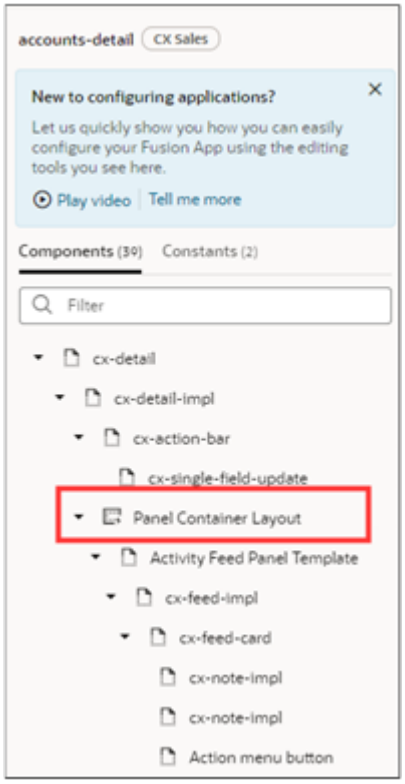
1. In Visual Builder Studio, click the **Page Designer** tab on the accounts-detail page.

   You can get to the accounts-detail page either by navigating to Visual Builder Studio from the Account detail page in the application, or by navigating to the accounts-detail node on the App UIs tab in Visual Builder Studio.



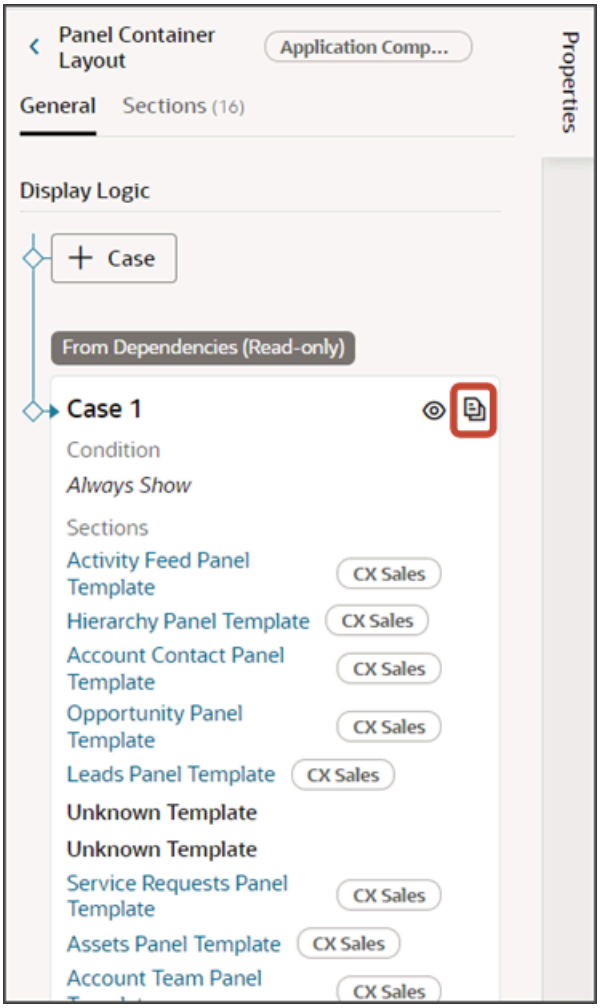
2. Click the **Design** button.

3. Duplicate the **Panel Container Layout** and add the panels to it:
    a. On the Structure panel, click the **Panel Container Layout** node.

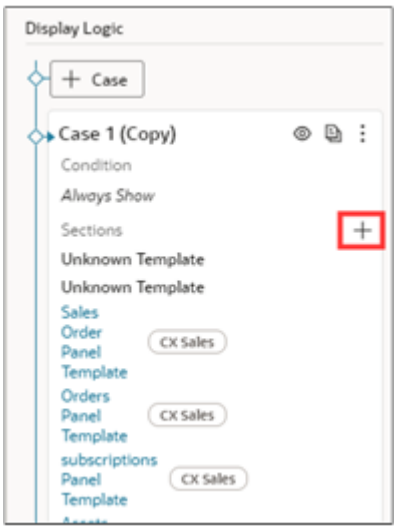


    b. On the Properties pane, next to **Case 1**, click the **Duplicate** icon.

    The container layout can include different variations of the layout that display to different audiences. For Accounts, there's a different layout for sales administrators, for sales managers, and sales representatives. You can duplicate any of these layouts, or the Case 1 layout we're duplicating here.

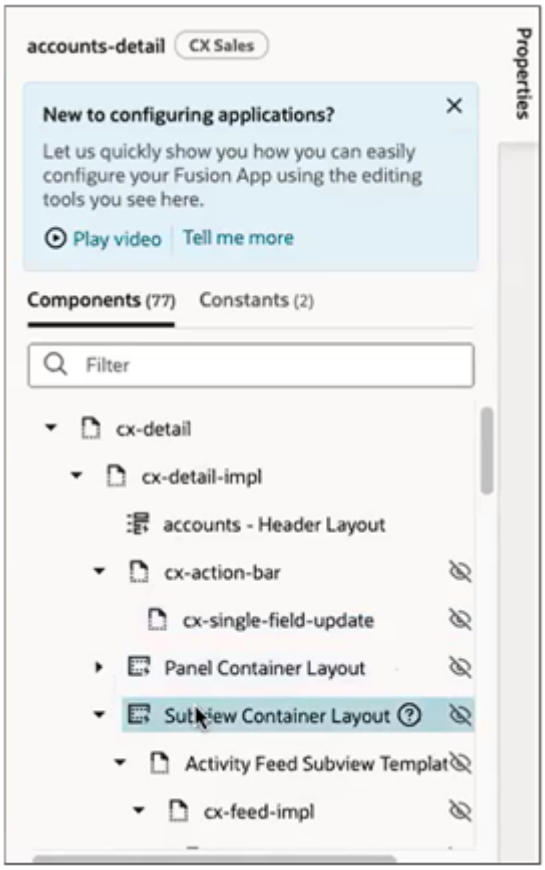


c. In the Case 1 (Copy), click **Add Panel** (the plus icon next to the Sections heading highlighted in the screenshot).



d. Click the panel template for the object you've added. In the example, that's the Shipment Panel Template.

4. Now repeat the same process for the **Subview Container Layout**.



   a. On the Structure panel, click the **Subview Container Layout** node.
   b. On the Properties pane, next to **Case 1**, click the **Duplicate** icon.
   c. In the Case 1 (Copy), click **Add Panel** (the plus icon next to the Sections heading ..
   d. Click the panel template for the object you've added. In the example, that's the **Shipment Subview Template**.
5. You can preview your configuration.
6. Publish the sandbox to enable the Smart Actions that were created in the CX Extension Generator.

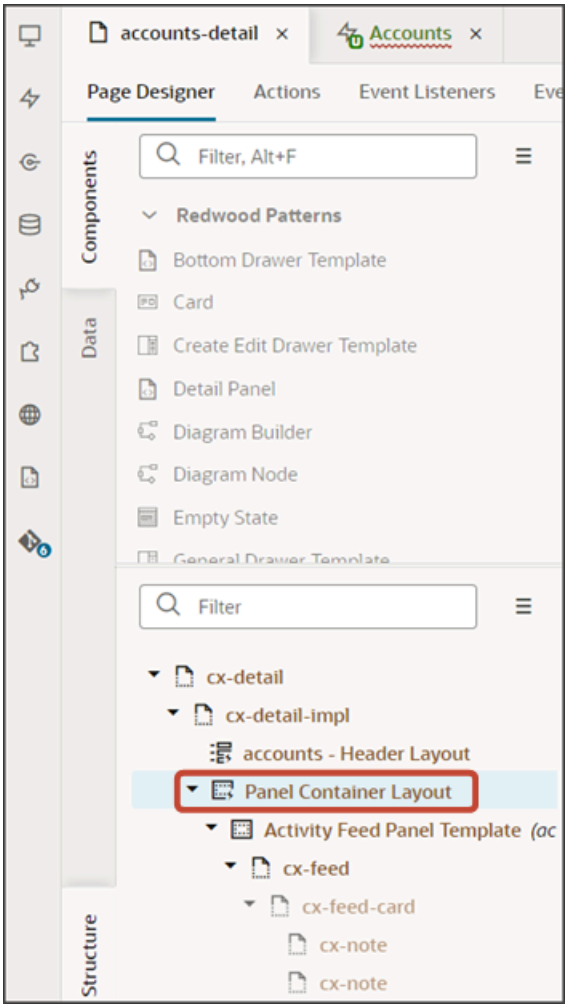
Change the Display Order of Panels in a Detail Page

You can change the display order of panels using Oracle Visual Builder Studio. Here's how.

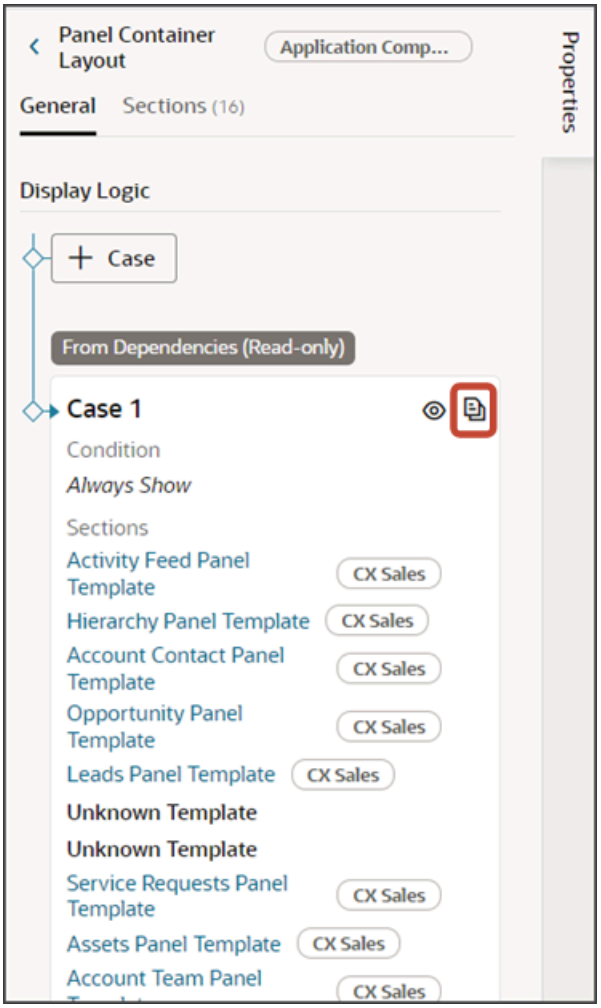
In this example, we'll move the Activities panel for an account so that it displays after the Hierarchy panel.

1. In Visual Builder Studio, navigate to the Page Designer tab on the accounts-detail page.
2. Click the **Design** button.

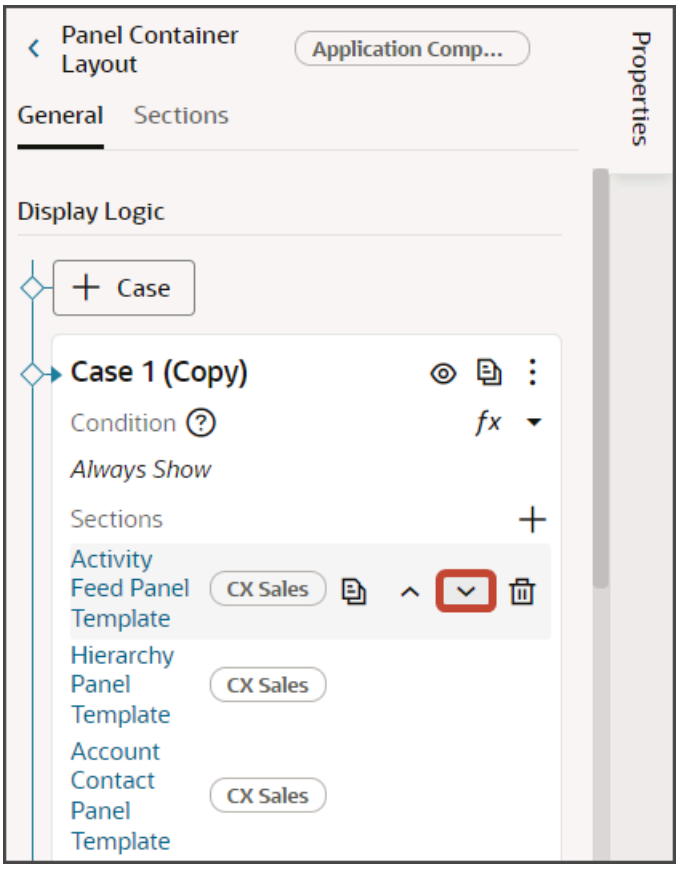
3. On the Structure panel, click the **Panel Container Layout** node.


accounts-detail
Accounts
Page Designer
Actions
Event Listeners
Components
Filter, Alt+F
Redwood Patterns
Bottom Drawer Template
Card
Create Edit Drawer Template
Detail Panel
Diagram Builder
Diagram Node
Empty State
Data
Filter
cx-detail
cx-detail-impl
accounts - Header Layout
Panel Container Layout
Activity Feed Panel Template
cx-feed
cx-feed-card
cx-note
cx-note
Structure

4. On the Properties pane, next to **Case 1**, click the Duplicate icon.

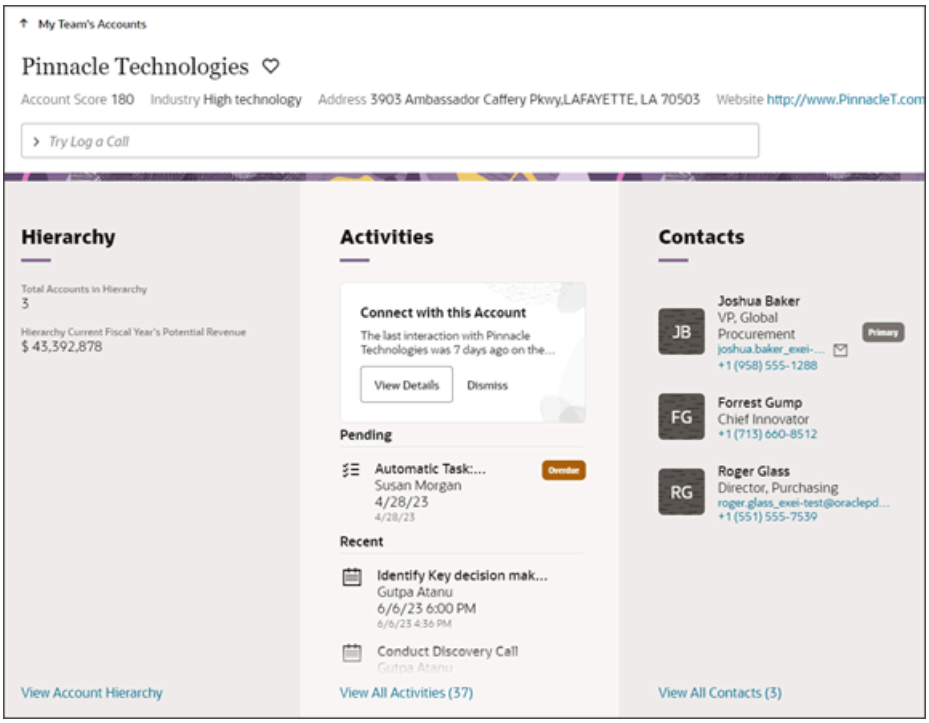


5. On the Properties pane for the copied layout, next to the Activity Feed Panel Template section, click the Move Down icon so that this section displays after the Hierarchy Panel Template section.



6. Click the Preview button to see your changes in your runtime test environment.

Here's a screenshot of the Activities panel which now displays right after the Hierarchy panel.



Display a Panel Based on a User's Role

To display a panel based on a user's role, you have two options: one option for a simple hide/show of a panel and a second option for more complicated scenarios such as reordering panels.

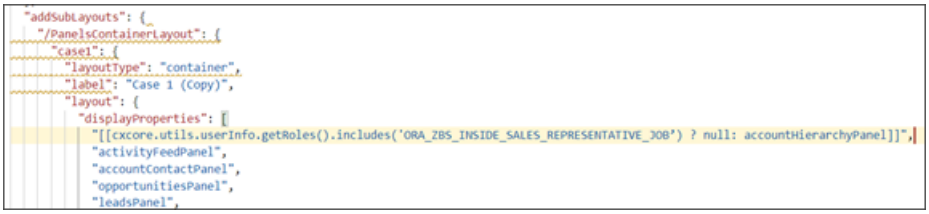
First, let's configure the panel region so that the Hierarchy panel displays only to sales managers, not sales representatives. This example is a simple hide/show use case, so it's easiest to update the accounts-detail page's JSON with the condition.

In the accounts-detail page's JSON, indicate that the Hierarchy panel must not display for sales representatives.

1. In Visual Builder Studio, click the App UIs side tab.
2. Navigate **CX Sales** > **cx-sales** > **accounts** > **accounts-detail**.
3. On the accounts-detail page, click the JSON subtab.
4. Scroll to the "/PanelsContainerLayout" section.

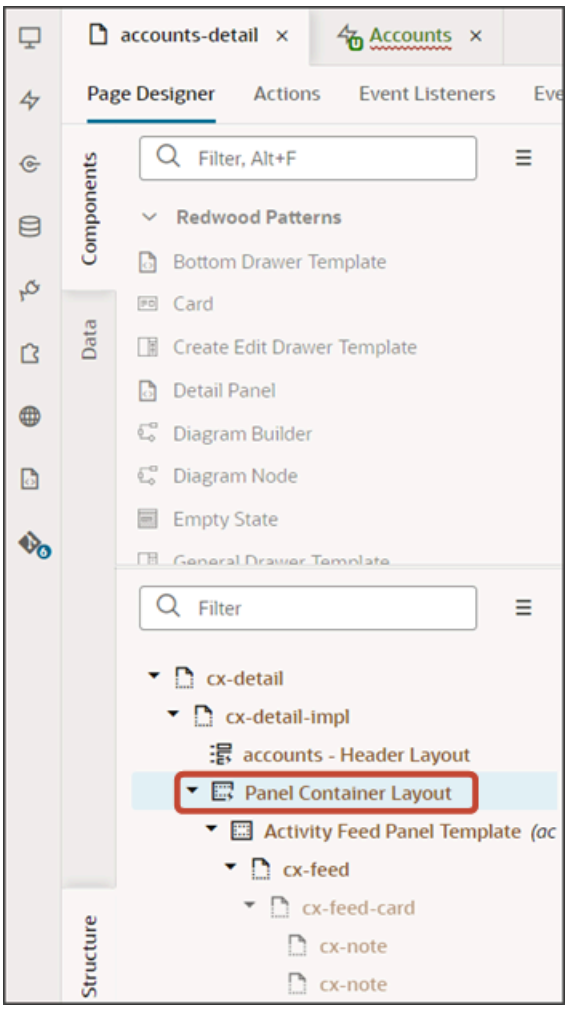
5. Replace the "accountHierarchyPanel" panel with the following expression:

```
"[[cxcore.utils.userInfo.getRoles().includes('ORA_ZBS_INSIDE_SALES_REPRESENTATIVE_JOB') ? null:
accountHierarchyPanel]]",
```

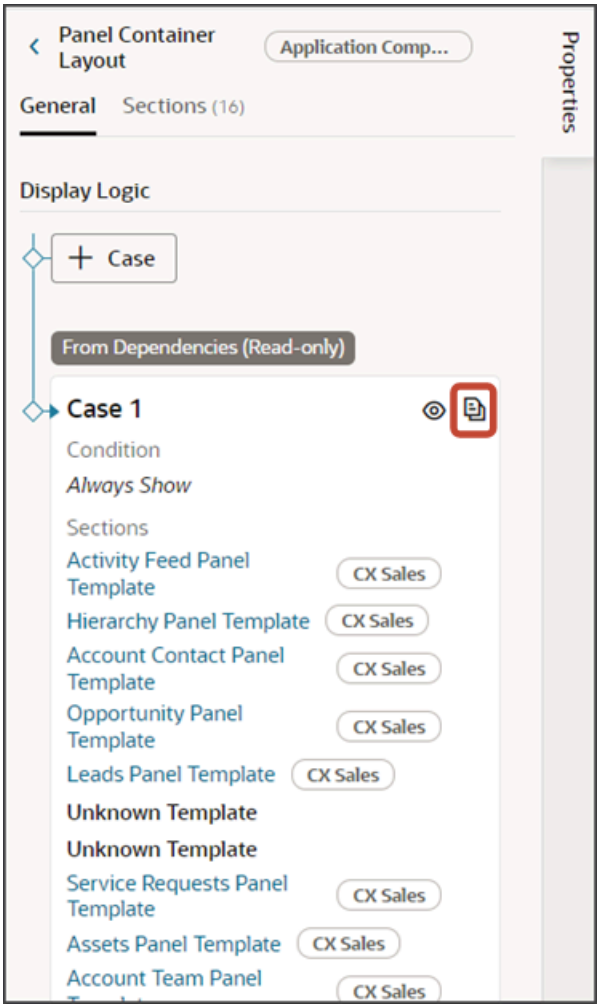


For more complex changes like reordering panels based on a user's role, you must create a panel layout with the new sequence and add a condition. For example, for the Case 1 (Copy) panel layout that we previously created, add the sales manager condition. This means that only sales managers can view the panel layout where the Activities panel displays after the Hierarchy panel.

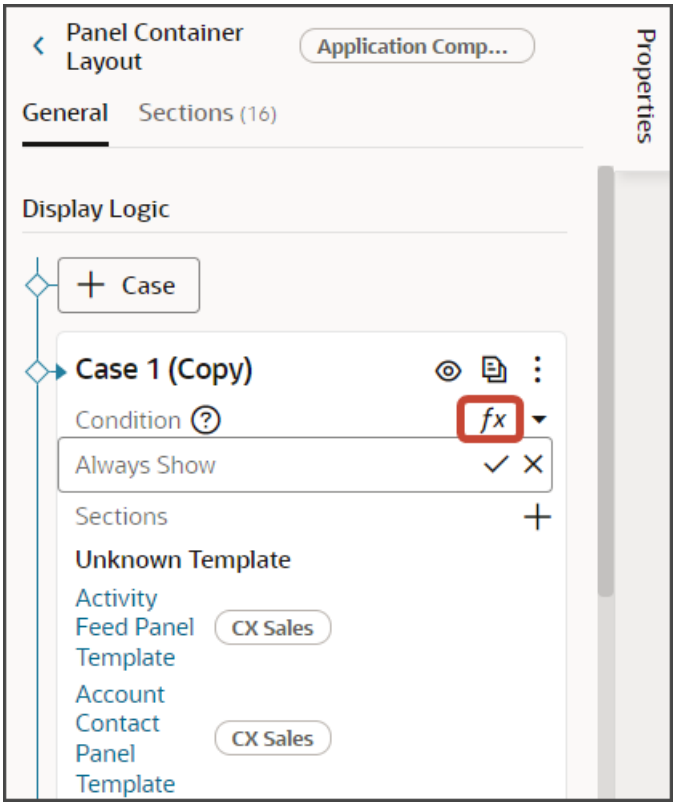
1. Navigate to the Page Designer tab on the accounts-detail page.
2. On the Structure panel, click the **Panel Container Layout** node.



3. On the Properties pane, next to **Case 1**, click the Duplicate icon.

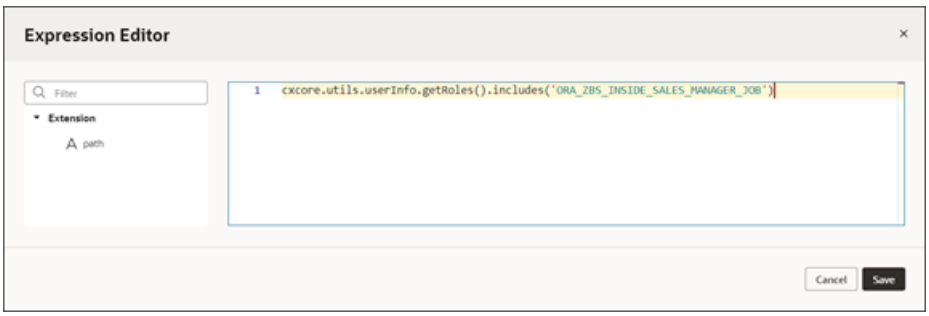


4. Next to the Case 1 (Copy) panel layout's condition, click the Expression Editor icon.



5. In the Expression Editor dialog, enter this expression:

```
cxcore.utils.userInfo.getRoles().includes('ORA_ZBS_INSIDE_SALES_MANAGER_JOB')
```



6. Click **Save**.

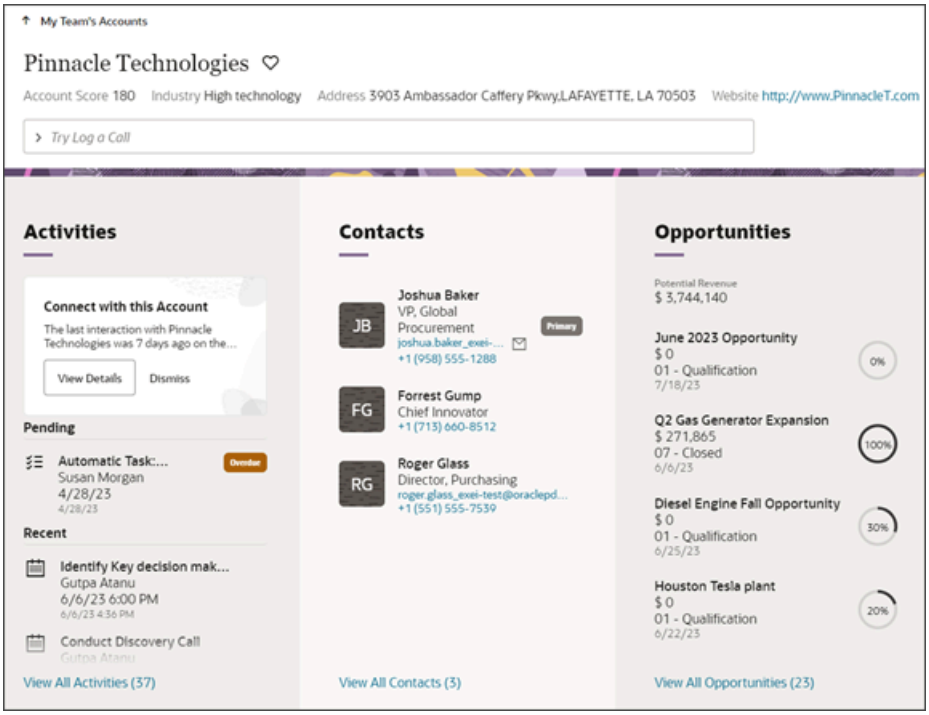
7. Let's test the sales representative change.

   Click the Preview button to see your changes in your runtime test environment.

   The preview link must include the `application/container` segments in the URL. If not, then change the preview link using the following example URL:

```
https://<servername>/fscmUI/redwood/cx-sales/application/container/accounts/accounts-detail?id=300000003513233&puid=7050&view=foldout
```

   Here's a screenshot where the logged-in user is a sales representative. Notice that, on the account detail page, the Hierarchy panel doesn't display. This is due to the JSON update that you made previously.



   Next, try signing in as a sales manager. On an account detail page, the Activities panel should display after the Hierarchy panel.

Display a Panel and Subview Based on a Field Value

You can display different sets of panels (and their corresponding subviews) based on the value of a field.

To do this, create a panel layout or subview layout, and then add a field value condition. If a record's field matches the specified value, then the associated layout displays. If not, then a different layout displays.

This topic illustrates how an account's type, either Customer or Prospect, changes the panel and subview layout on an account detail page.

Prerequisite

To create a layout condition that references a field value, you must first enable this feature so that panels and subviews are loaded to the page only after evaluating the header.

1. In Visual Builder Studio, click the App UIs side tab.
2. Navigate **CX Sales** > **cx-sales** > **accounts** > **accounts-detail**.
3. On the accounts-detail page, click the Variables subtab.
4. In the Constants region, click the **deferRelatedDataLoad** constant.
5. On the Properties pane, in the Default Value field, select **True**.

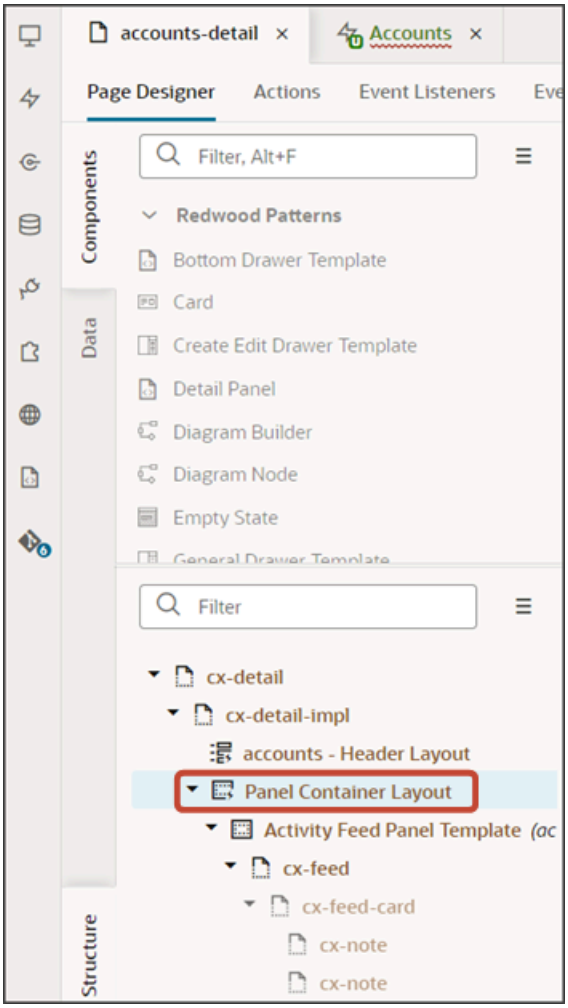
   If you want to add a field value condition to panel and subview container layouts, then you must set this value to true.

Create a New Panel Layout

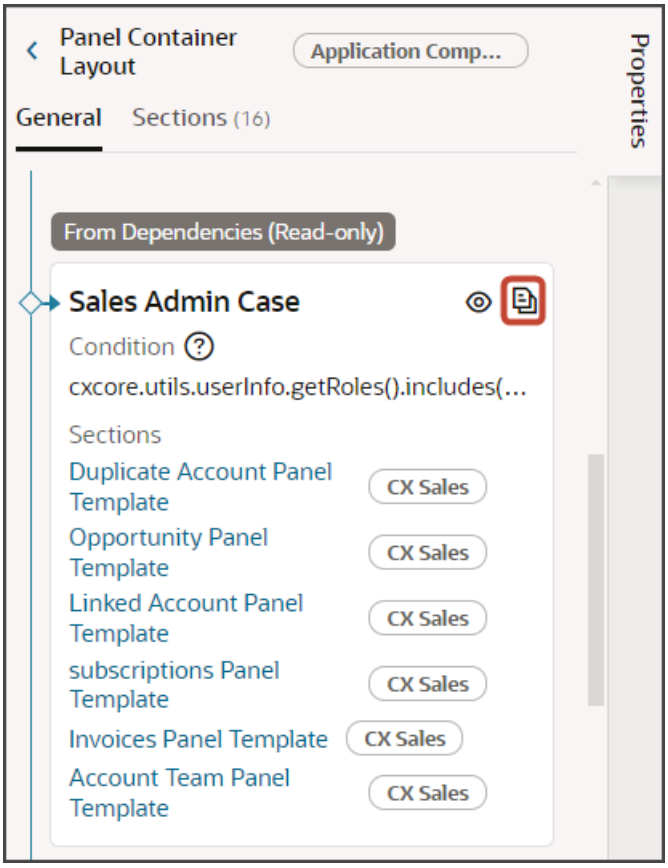
Once you have enabled the feature, you can now add a field value condition to a panel layout. Let's add a condition to the account detail page.

1. Navigate to Visual Builder Studio from an account record.
2. On the accounts-detail page, click the Page Designer subtab.

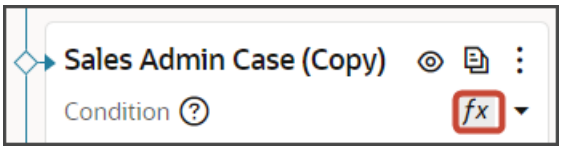
3. On the Structure panel, click the **Panel Container Layout** node.



4. On the Properties pane, next to **Sales Admin Case**, click the Duplicate icon.



5. Next to the Sales Admin Case (Copy) panel layout's condition, click the Expression Editor icon.



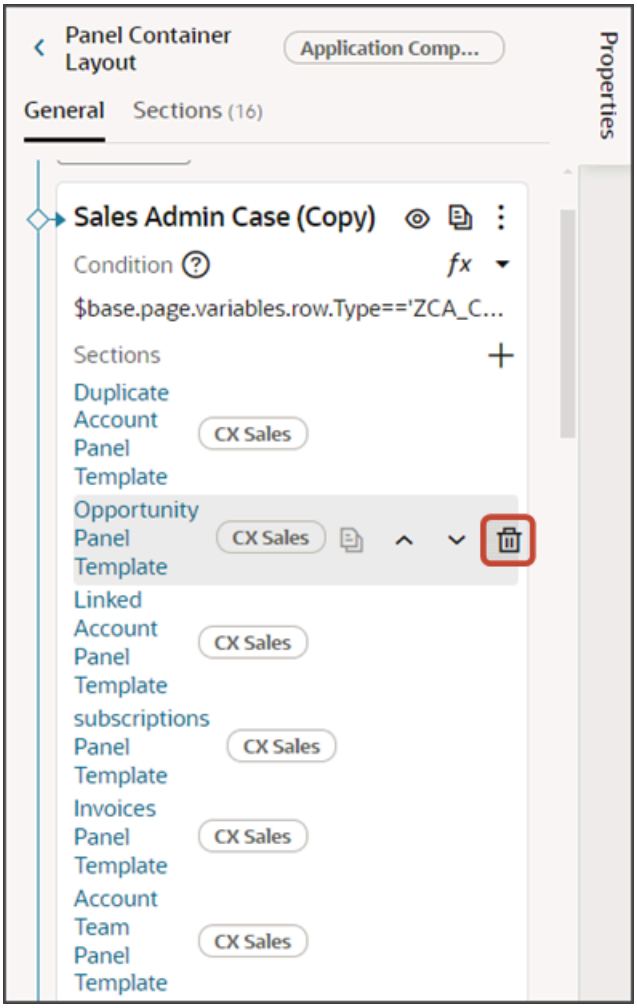
6. In the Expression Editor dialog, replace the existing expression with this new one, just for testing:

```
$base.page.variables.row.Type=='ZCA_CUSTOMER'
```

7. Click **Save**.

8. Delete the Opportunity Panel Template.

   With the field condition specified above, this means that accounts of type Customer won't see the Opportunities panel on the account detail page.



9. Let's test this change.

   From the accounts-list page, click the Preview button to see your changes in your runtime test environment.

10. The resulting preview link will be:

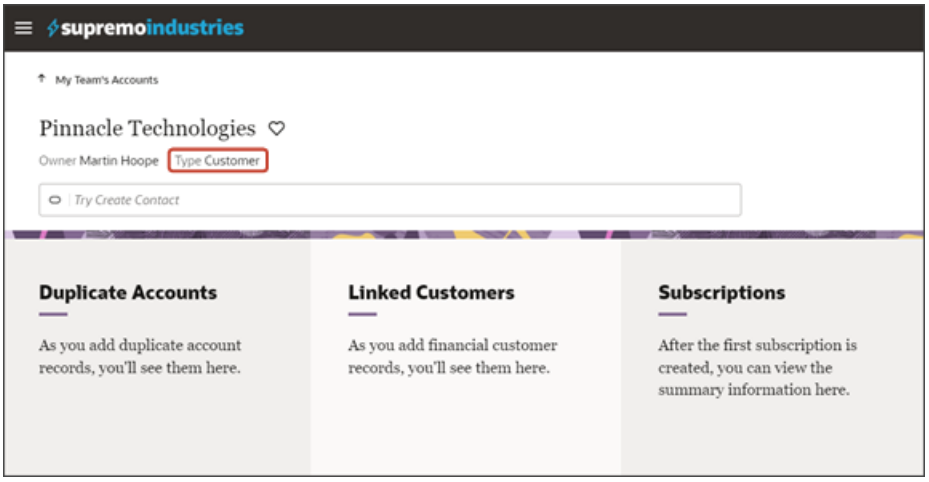
    ```
    https://<servername>/fscmUI/redwood/cx-sales/accounts/accounts-list
    ```

11. Change the preview link as follows:

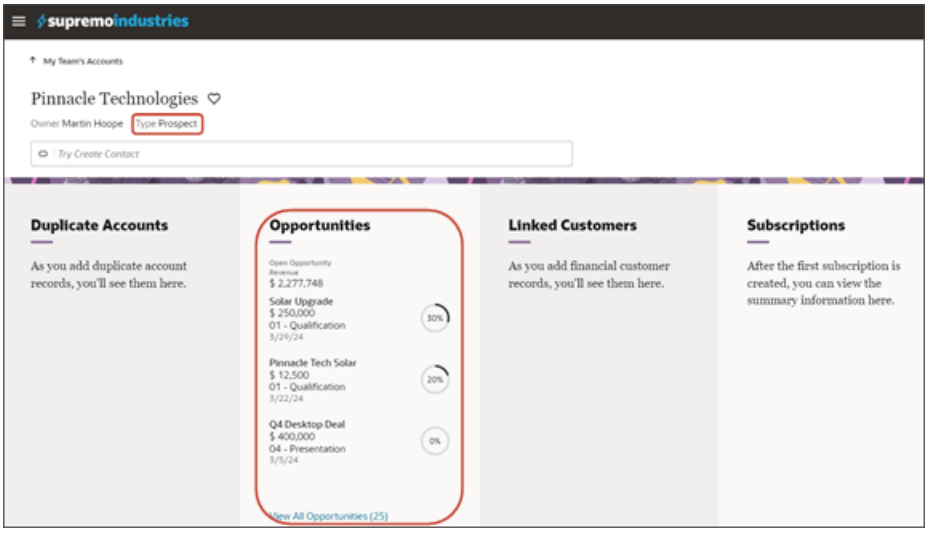
    ```
    https://<servername>/fscmUI/redwood/cx-sales/application/container/accounts/accounts-list
    ```

    Note: You must add `/application/container` to the preview link.

12. On the My Team's Accounts page, click any account.
    - If the account is of type Customer, then you won't see the Opportunities panel.

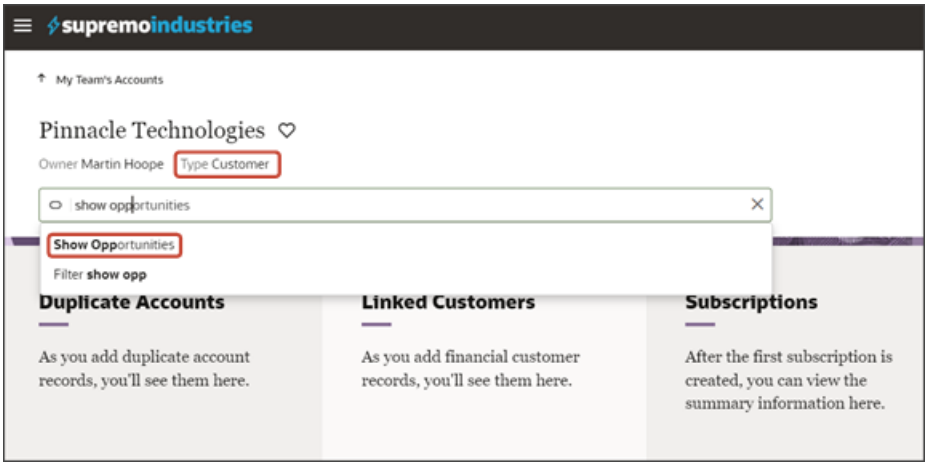


    - If the account is of type Prospect, then the Opportunities panel does display.



Create a New Subview Layout

Next, add the field condition to the subview layout, as well. It's important to add the field condition to the subview layout. Otherwise, the Show Opportunities smart action is still available from the Action Bar even when the account is a customer.



1. Navigate to Visual Builder Studio from any subview page, which you can navigate to from any panel on an account record.
2. On the accounts-detail page, click the Page Designer subtab.
3. On the Structure panel, click the **Subview Container Layout** node.
4. On the Properties pane, next to **Subview Container Layout**, click the Duplicate icon.
5. Next to the Subview Container Layout (Copy) subview layout's condition, click the Expression Editor icon.
6. In the Expression Editor dialog, add this expression:

   ```
   $base.page.variables.row.Type=='ZCA_CUSTOMER'
   ```
7. Click **Save**.
8. Delete the Opportunity Subview Template.

   With the field condition specified above, this means that accounts of type Customer won't see the Show Opportunities smart action on the account detail page.
9. Let's test this change.

   From the accounts-list page, click the Preview button to see your changes in your runtime test environment.

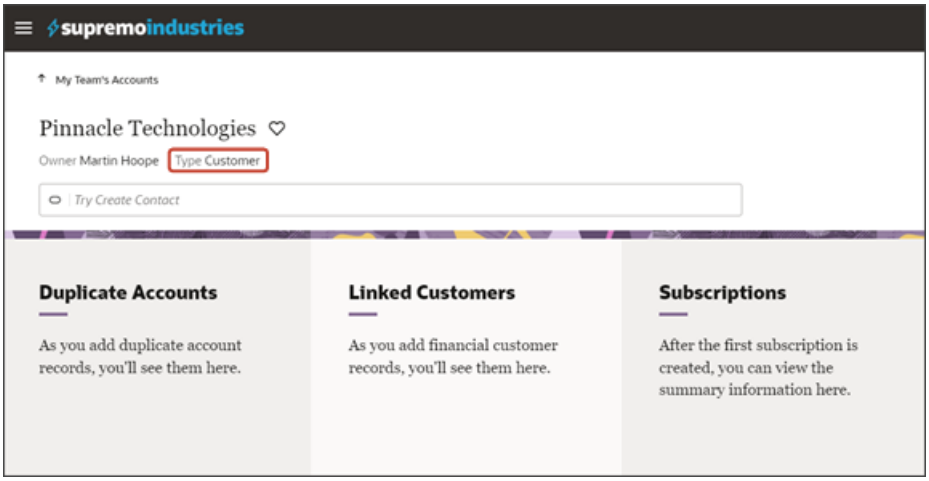
10. The resulting preview link will be:

    ```
    https://<servername>/fscmUI/redwood/cx-sales/accounts/accounts-list
    ```
11. Change the preview link as follows:

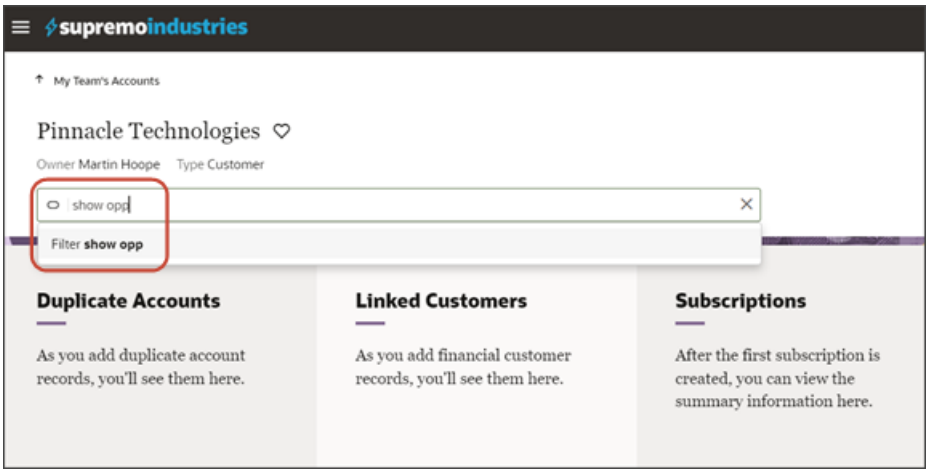
    ```
    https://<servername>/fscmUI/redwood/cx-sales/application/container/accounts/accounts-list
    ```

    Note: You must add `/application/container` to the preview link.

12. On the My Team's Accounts page, click any account and make sure that the account is of type Customer. The Opportunities panel shouldn't display.



13. Test the field condition on the subview layout by checking to see if the Show Opportunities smart action is still available from the Action Bar. It shouldn't be visible anymore if the account is a customer.

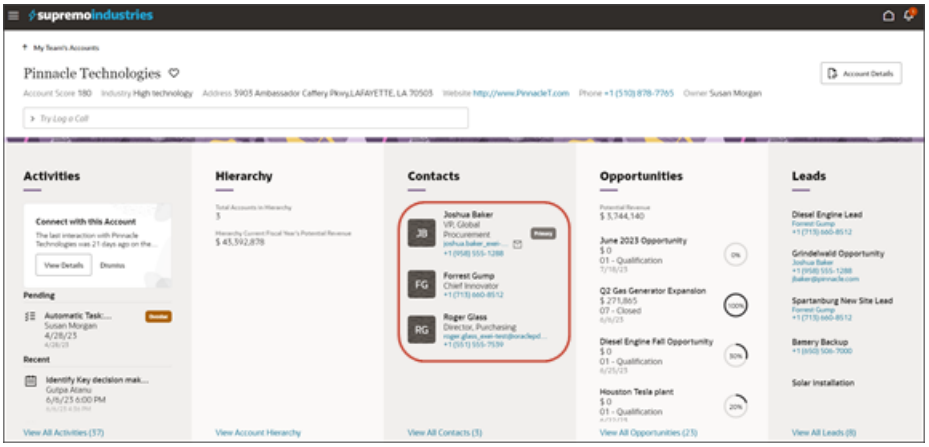


Configure the Contents of a Panel

An individual record's detail page includes key information displayed in a region of panels. Each panel contains information related to the record, such as related contacts and opportunities. Most panels display information in a list format. You can configure these lists using Oracle Visual Builder Studio.

What's Inside a Panel?

A panel often contains a list, which you can configure. Here's an example of a list inside a panel:



Lists can display up to 5 records, depending on screen size. If the screen size is small, then the list automatically adjusts to display fewer records. However, users can click the View All link that displays at the bottom of the panel to navigate to a second page to see all records in the list. This second page is called the subview.

What Can You Change in a List?

In Visual Builder Studio, you can modify the information that displays in each list.

You can:

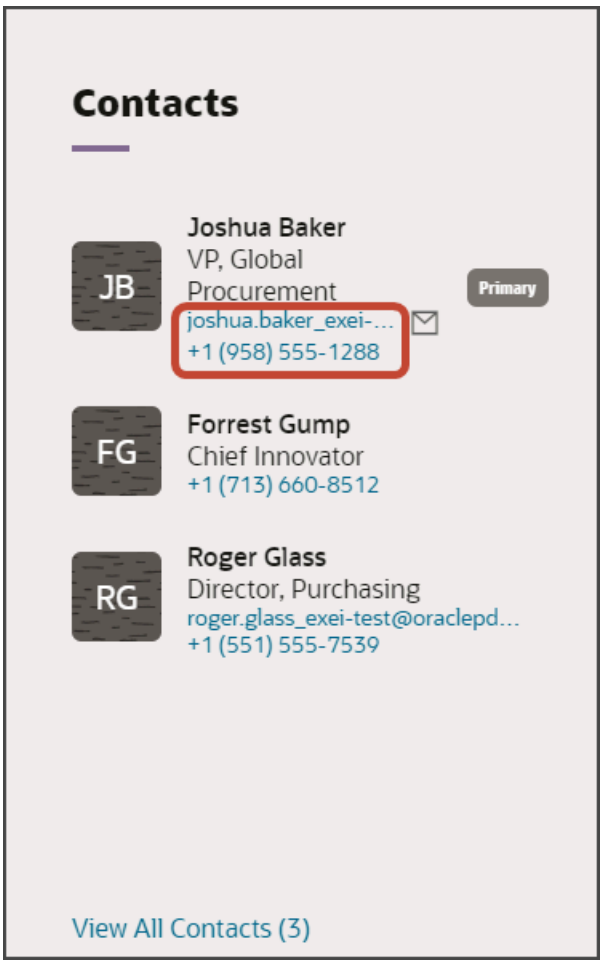
- Add and remove fields
- Change the display order of fields in the list

This topic illustrates how to change the display order of fields that display on panels on an account's detail page. We'll look at both the Contacts panel and Opportunities panel.

To configure the subview, see *Configure the Subview Layout*.

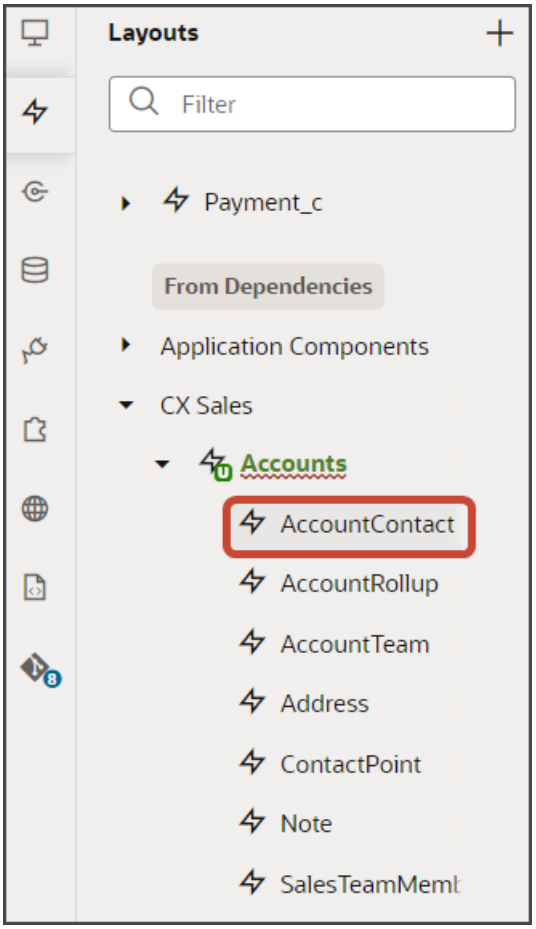
Change the Display Order of Contact Panel Fields

Let's change the display order of fields in a panel list. In this example, we'll switch the order of the email and phone number fields on the Contacts panel on the Account detail page.



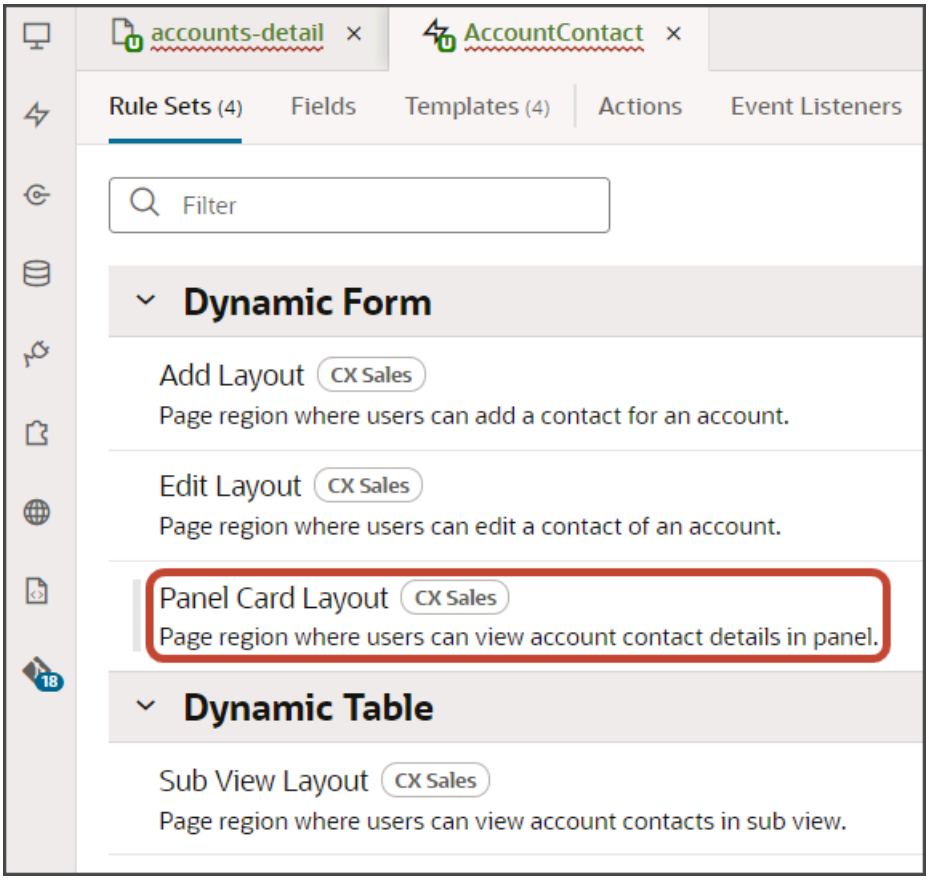
1. In Visual Builder Studio, navigate to the Layouts tab and expand the CX Sales node > Accounts > AccountContact.

   The AccountContact node contains the rule sets for the Contacts panel on the Account object.

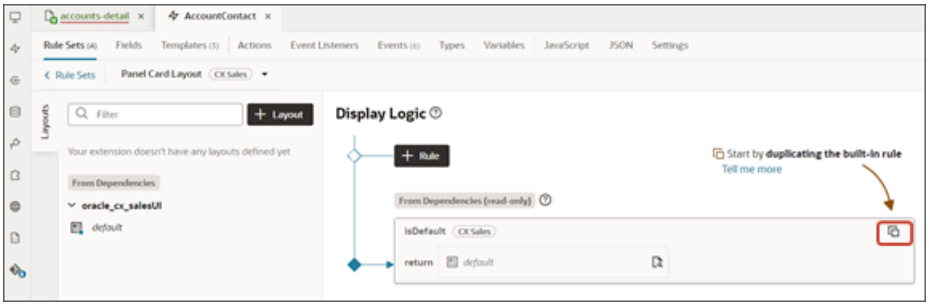


Note: When configuring the contents of a panel, consider what kind of relationship the panel's object has with the primary object. In this case, the Account object has a many-to-many relationship with Contact. This means that you'll find layouts for the Contact object on the AccountContact node, nested under the Accounts node.

2. On the AccountContact tab > Rule Sets subtab, click the Panel Card Layout.



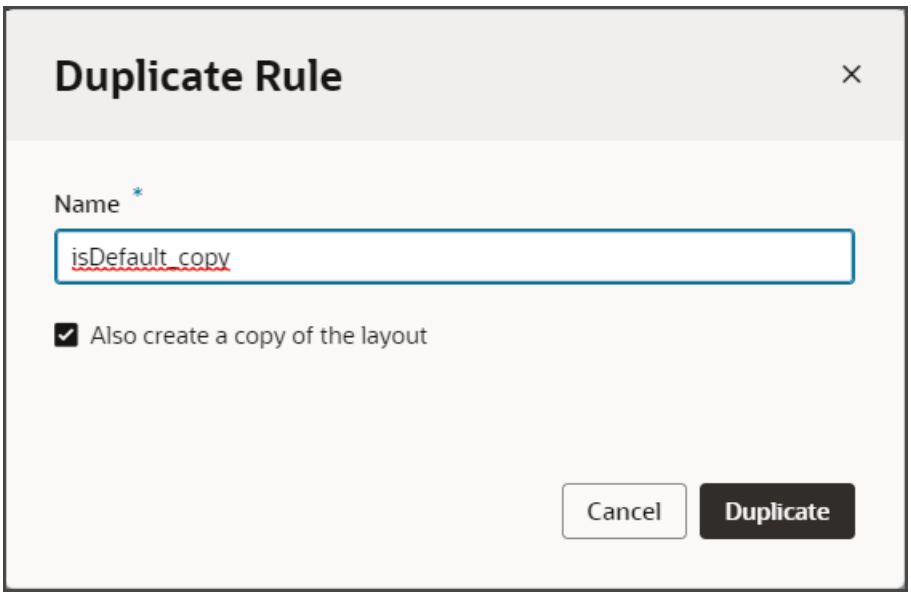
Both a default layout as well as a default rule are displayed for the Panel Card Layout.



3. Click the Duplicate Rule icon.

4. In the Duplicate Rule dialog, accept the default rule name or enter a new name. The name you enter here is both the rule name and also the layout name, so enter a layout name that makes sense for you.

   Also, make sure that the **Also create a copy of the layout** check box is selected.



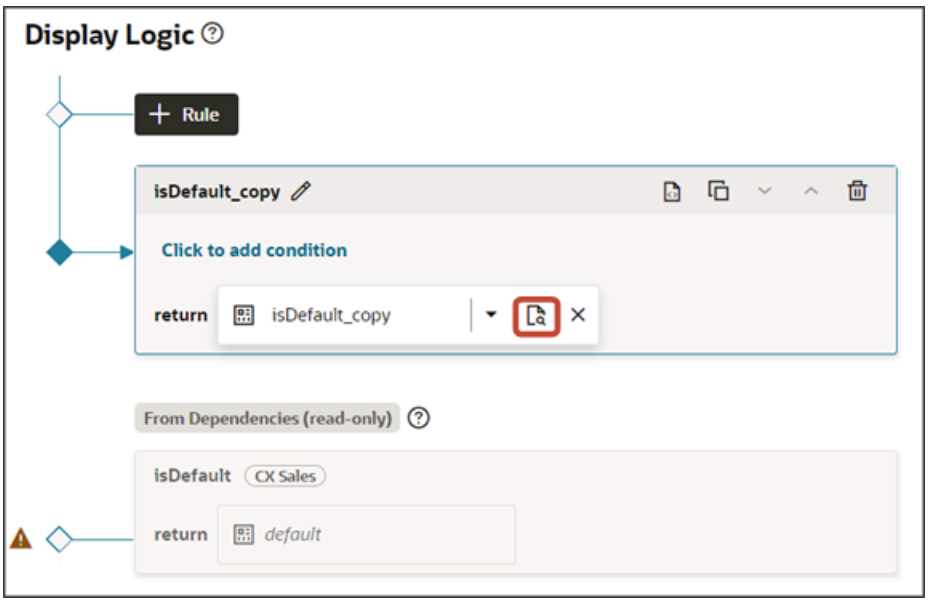
5. Click **Duplicate**.

   The new rule displays at the top of the list of existing rules, which means that this rule will be evaluated first at runtime. If the rule's conditions are met, then the associated layout is displayed to the user.

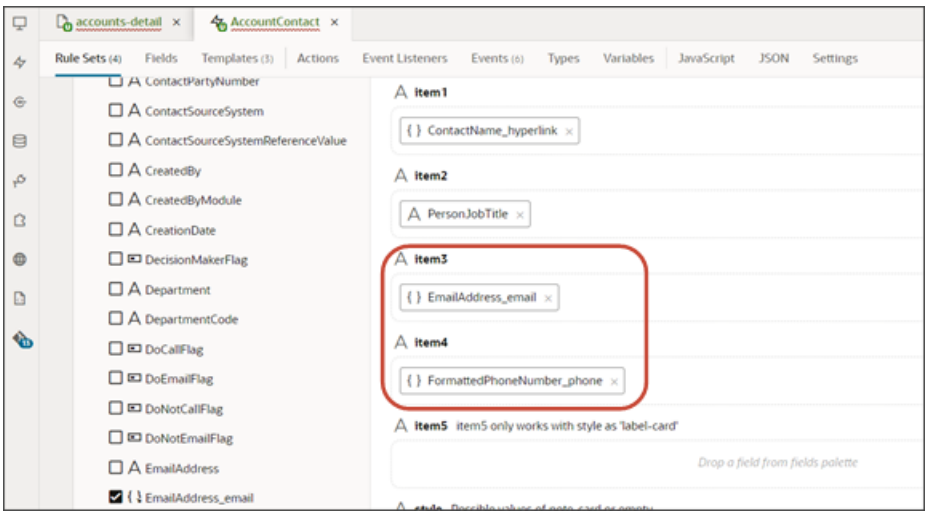
   In this example, we're not adding any conditions which means that the associated layout will always be displayed.

6. Modify the rule's copied layout.

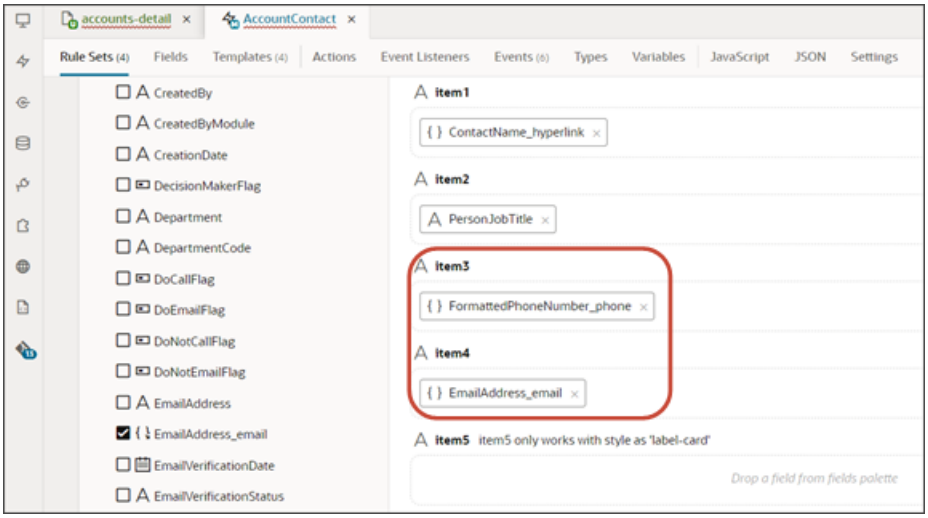
   a. Click the Open icon to edit the copied layout.



   b. Scroll down the list of fields in the layout until you locate the email and phone fields.



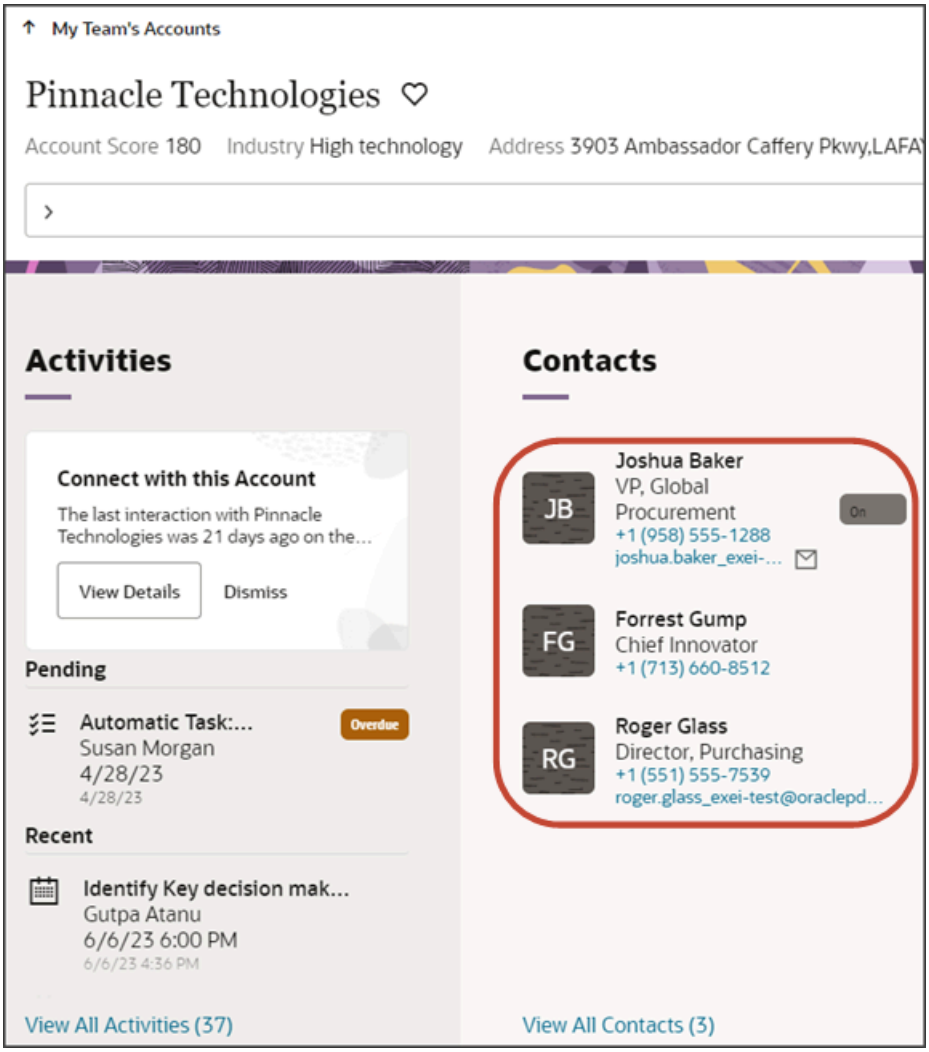
   c. Delete each field from the Item3 and Item4 slots, and then add the fields back. This time, however, switch the order so that the phone field is in the Item3 slot and the email field is in the Item4 slot.


accounts-detail
AccountContact
Rule Sets (4)
Fields
Templates (4)
Actions
Event Listeners
Events (6)
Types
Variables
JavaScript
JSON
Settings
CreatedBy
CreatedByModule
CreationDate
DecisionMakerFlag
Department
DepartmentCode
DoCallFlag
DoEmailFlag
DoNotCallFlag
DoNotEmailFlag
EmailAddress
EmailAddress_email
EmailVerificationDate
EmailVerificationStatus
item1
ContactName_hyperlink
item2
PersonJobTitle
item3
FormattedPhoneNumber_phone
item4
EmailAddress_email
item5 item5 only works with style as 'label-card'
Drop a field from fields palette

7. Click the Preview button to see your changes in your runtime test environment.

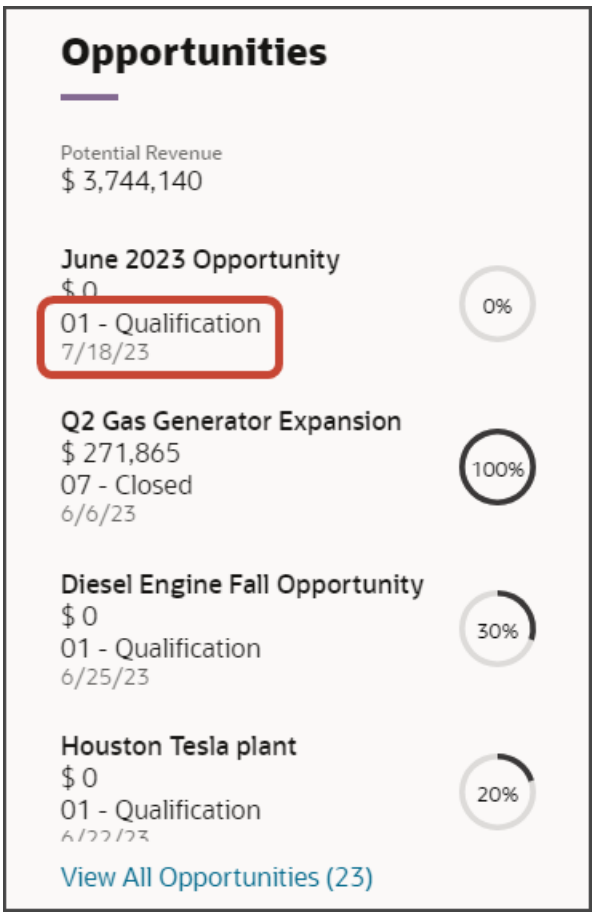
The preview link must include the `application/container` segments in the URL. If not, then change the preview link using the following example URL:

```
https://<servername>/fscmUI/redwood/cx-sales/application/container/accounts/accounts-detail?id=300000003513233&puid=7050&view=foldout
```



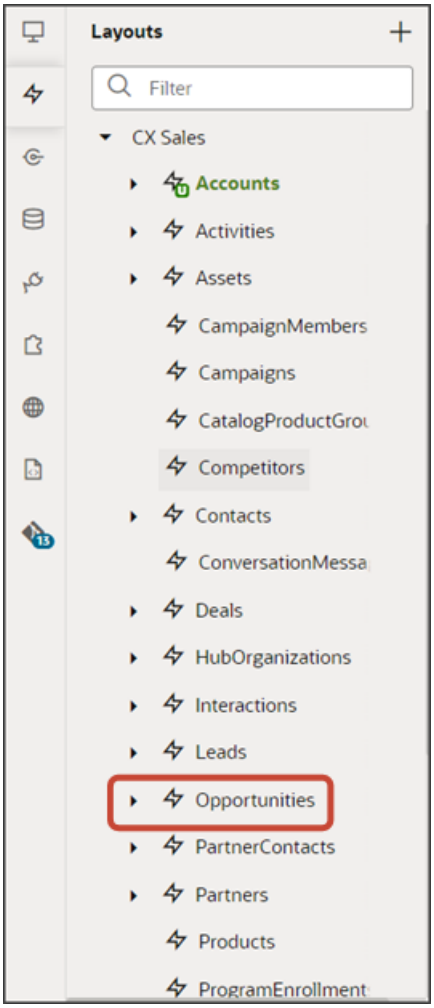
Change the Display Order of Opportunity Panel Fields

In this example, we'll switch the order of the sales stage and effective date fields on the Opportunities panel on the Account object.



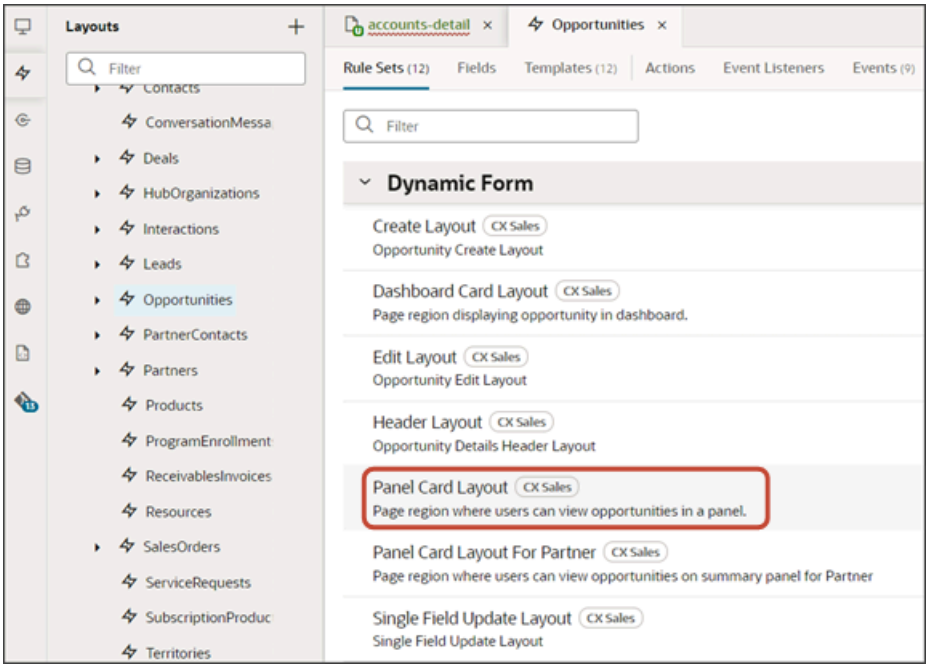
1. In Visual Builder Studio, navigate to the Layouts tab and expand the CX Sales node > Opportunities.

   The Opportunities node contains the rule sets for the Opportunities panel on the Account object.

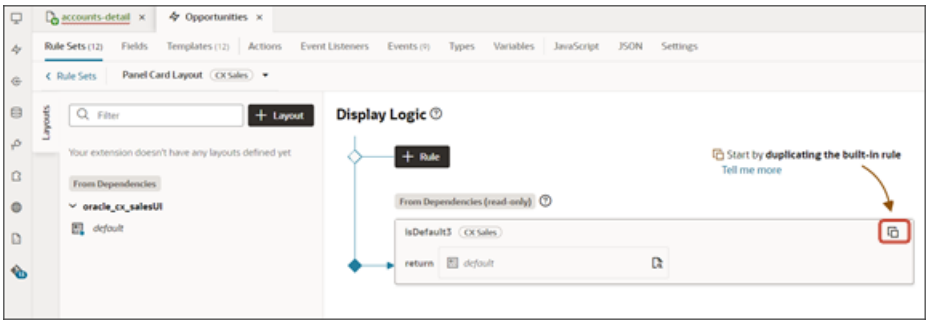


Note: When configuring the contents of a panel, consider what kind of relationship the panel's object has with the primary object. In this case, the Account object has a one-to-many relationship with Opportunity. This means that you'll find layouts for the Opportunity object on the Opportunities node.

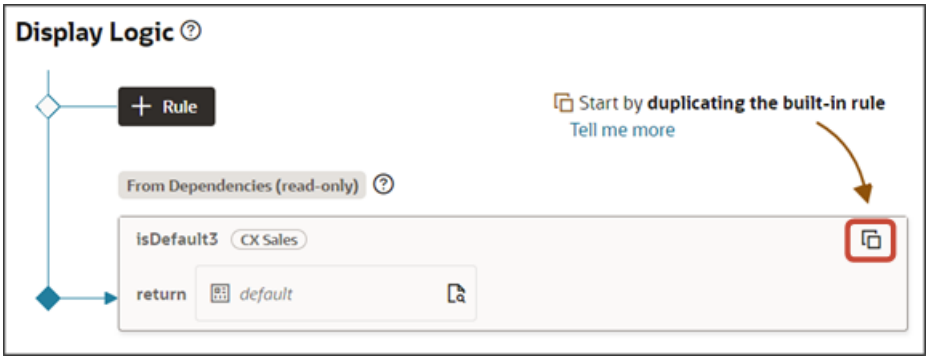
2. On the Opportunities tab > Rule Sets subtab, click the Panel Card Layout.



Both a default layout as well as a default rule are displayed for the Panel Card Layout.

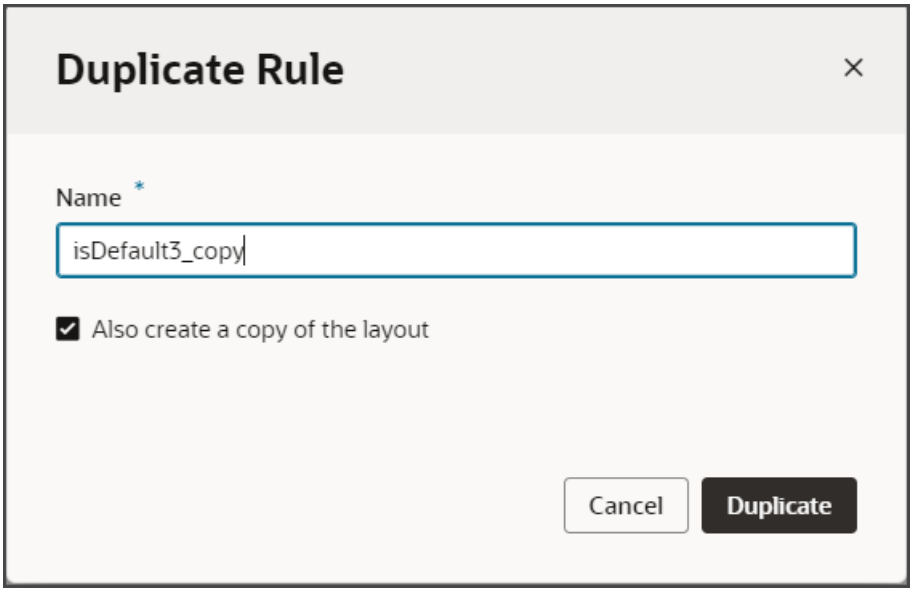


3. Click the Duplicate Rule icon.



4. In the Duplicate Rule dialog, accept the default rule name or enter a new name. The name you enter here is both the rule name and also the layout name, so enter a layout name that makes sense for you.

   Also, make sure that the **Also create a copy of the layout** check box is selected.



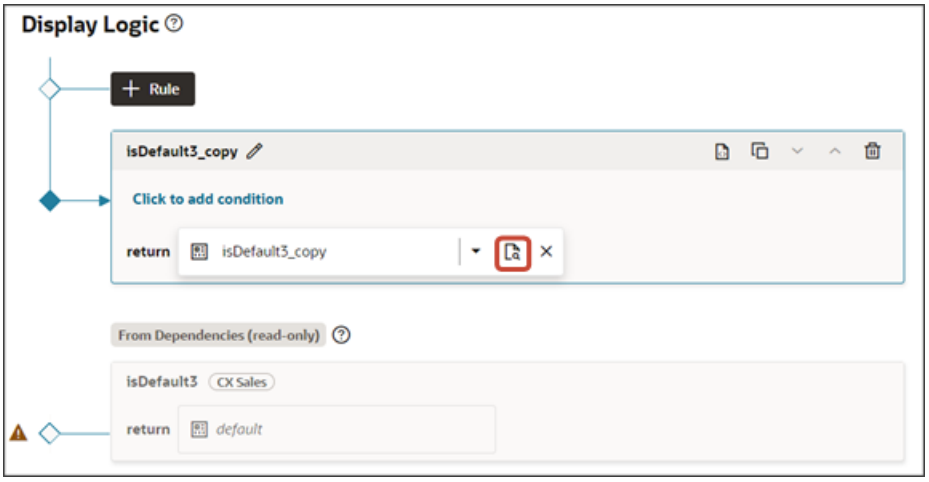
5. Click **Duplicate**.

   The new rule displays at the top of the list of existing rules, which means that this rule will be evaluated first at runtime. If the rule's conditions are met, then the associated layout is displayed to the user.

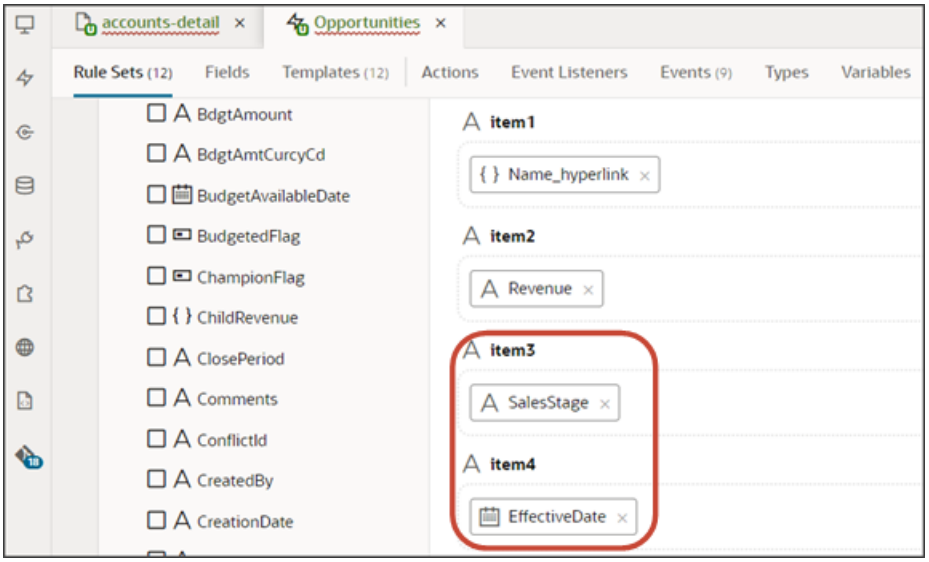
   In this example, we're not adding any conditions which means that the associated layout will always be displayed.

6. Modify the rule's copied layout.

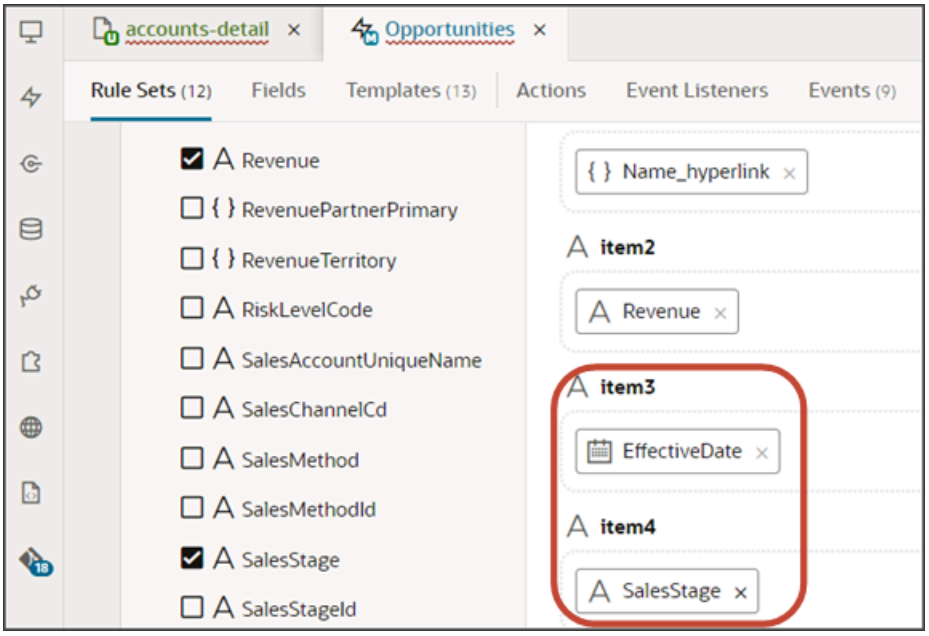
   a. Click the Open icon to edit the copied layout.



   b. Scroll down the list of fields in the layout until you locate the sales stage and effective date fields.



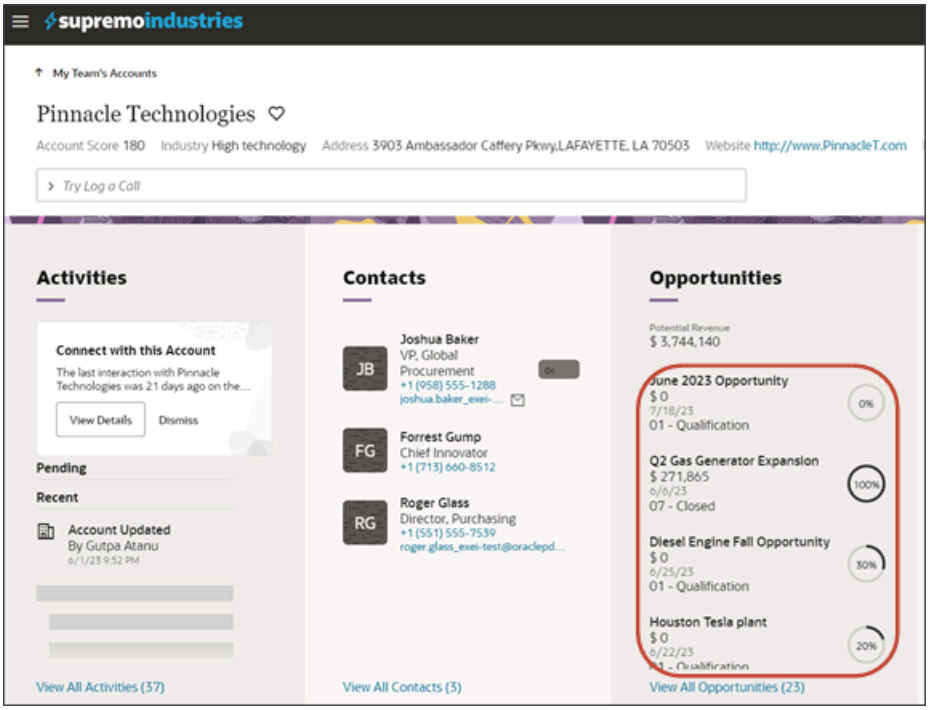
   c. Delete each field from the Item3 and Item4 slots, and then add the fields back but switch the order.


accounts-detail
Opportunities
Rule Sets (12)
Fields
Templates (13)
Actions
Event Listeners
Events (9)
Revenue
RevenuePartnerPrimary
RevenueTerritory
RiskLevelCode
SalesAccountUniqueName
SalesChannelCd
SalesMethod
SalesMethodId
SalesStage
SalesStageId
Name_hyperlink
item2
Revenue
item3
EffectiveDate
item4
SalesStage

7. Click the Preview button to see your changes in your runtime test environment.

The preview link must include the `application/container` segments in the URL. If not, then change the preview link using the following example URL:

```
https://<servername>/fscmUI/redwood/cx-sales/application/container/accounts/accounts-detail?
id=300000003513233&puid=7050&view=foldout
```



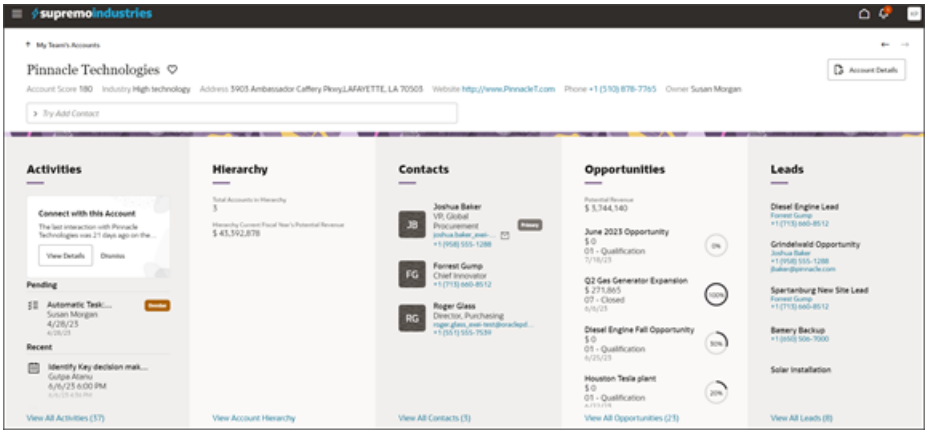
Configure the Subview Layout

An object's detail page includes a region of panels with information. Each panel, however, can display only a few records due to panel size. To see all records, users can navigate to a second page called a subview. This topic illustrates how to modify those subview pages using Oracle Visual Builder Studio.

What's Inside the Subview?

A subview contains a list of all records that the panel, due to limited real estate, can't display.

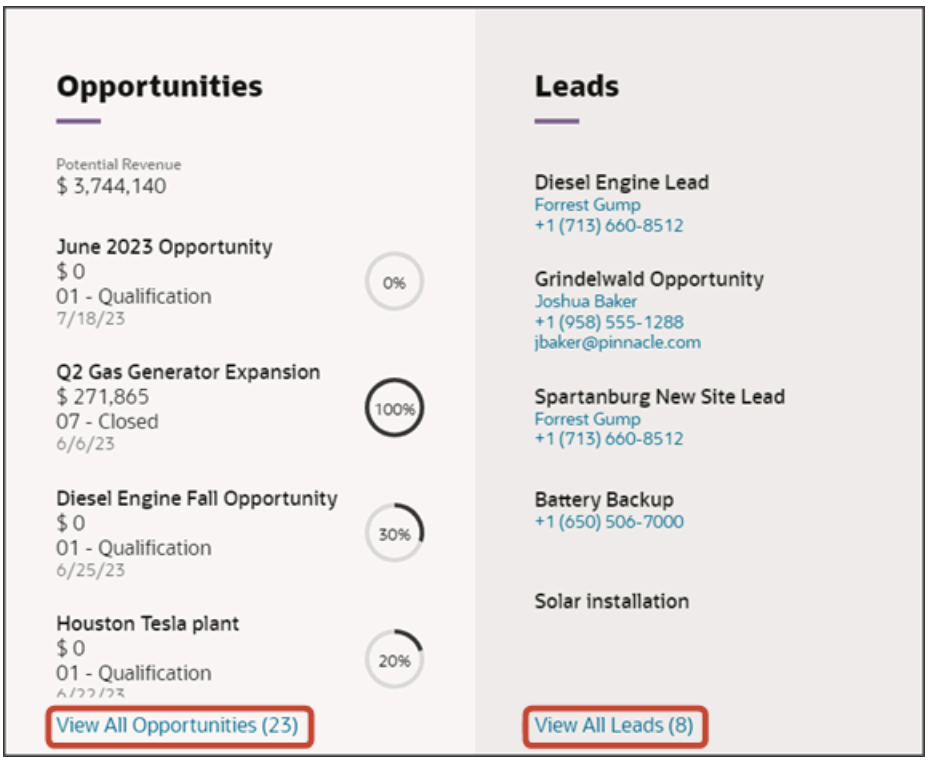
For example, here's an example of an account detail page with 5 panels:



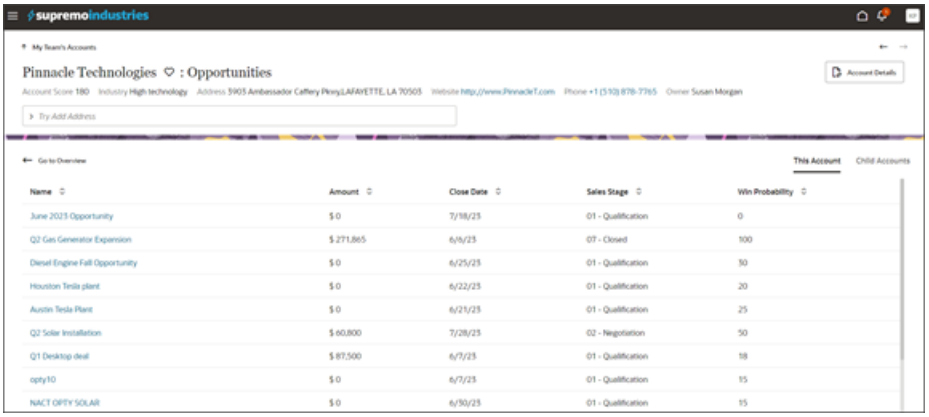
Notice how each panel displays only a few records.

To see all records, users can click the View All link that displays at the bottom of the panel.

Here's an example of some View All links. Note that after the link itself, a number indicates the number of total records listed on the subview.



The subview displays all those records in a table.



What Can You Change in a Subview Table?

In Visual Builder Studio, you can modify the information that displays in a subview table.

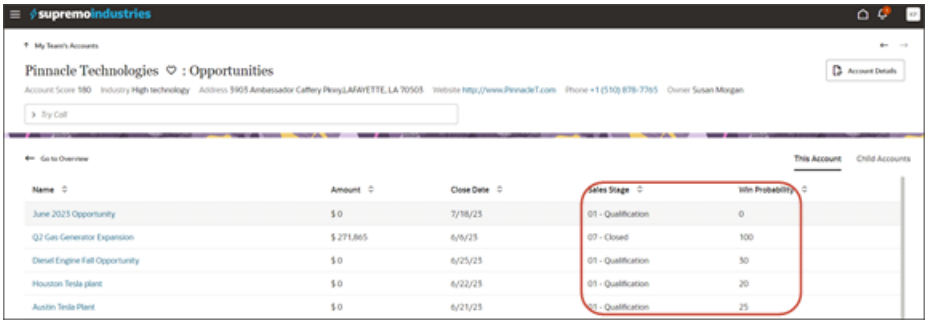
You can:

- Add and remove columns
- Change the display order of columns in the table

This topic illustrates how to change the display order of columns in a subview table. We'll look at the Opportunities subview that's available from an account detail page.

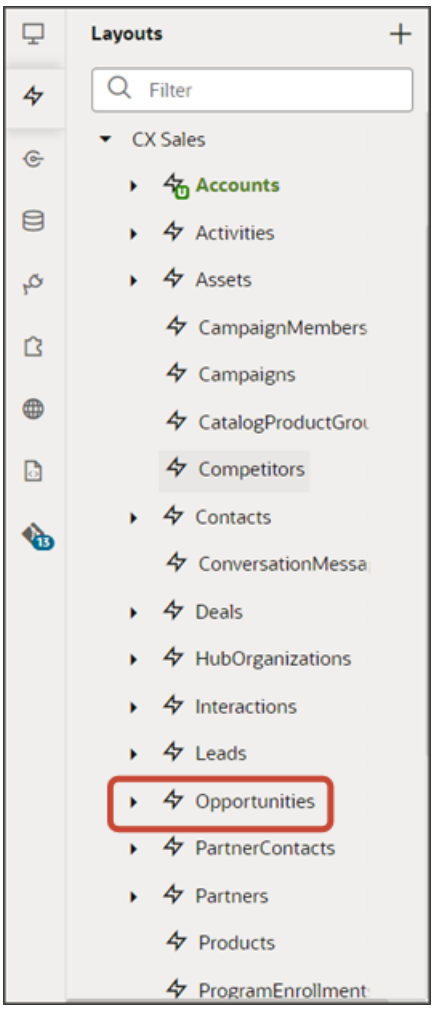
Change the Display Order of Opportunity Subview Columns

Let's change the display order of columns in a subview table. In this example, we'll switch the order of the sales stage and win probability columns on the Opportunities subview, accessed from the Opportunities panel on the Account detail page.

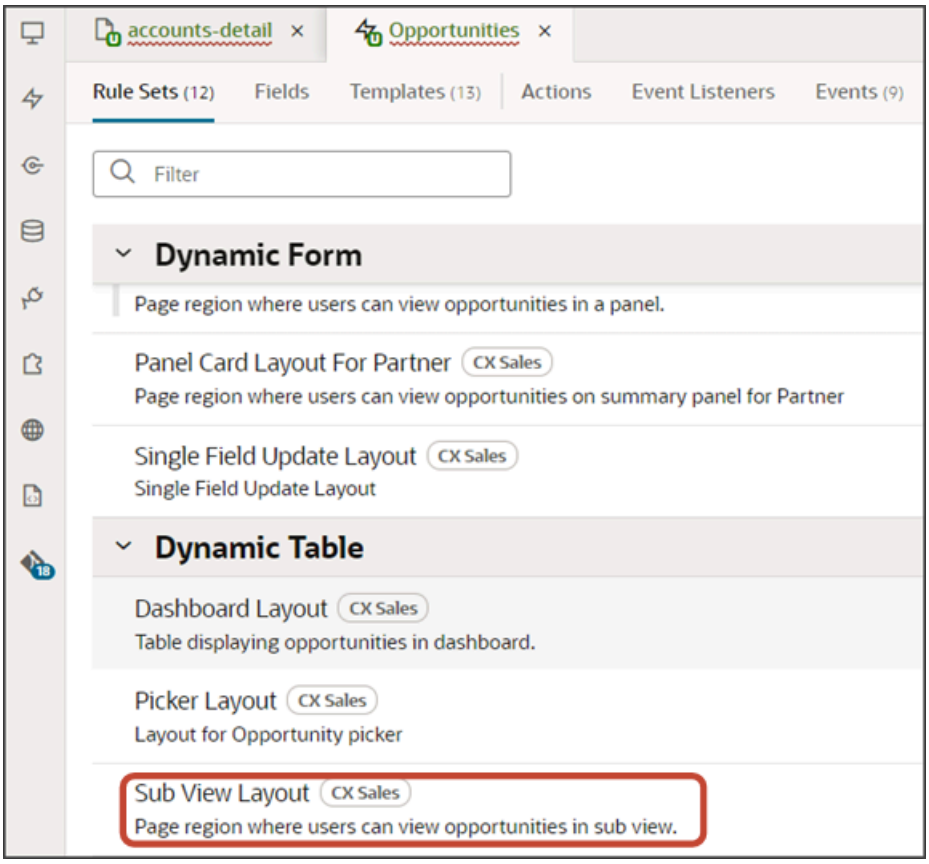


1. In Visual Builder Studio, navigate to the Layouts tab and expand the CX Sales node > Opportunities.

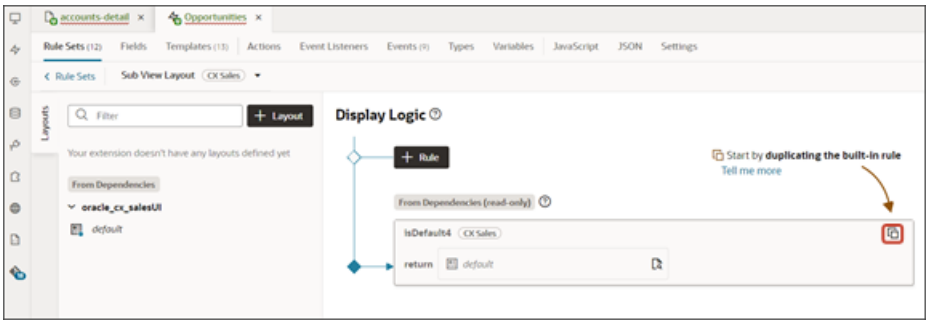
   The Opportunities node contains the rule sets for the Opportunities panel on the Account object.



2. On the Opportunities tab > Rule Sets subtab, click the Sub View Layout.



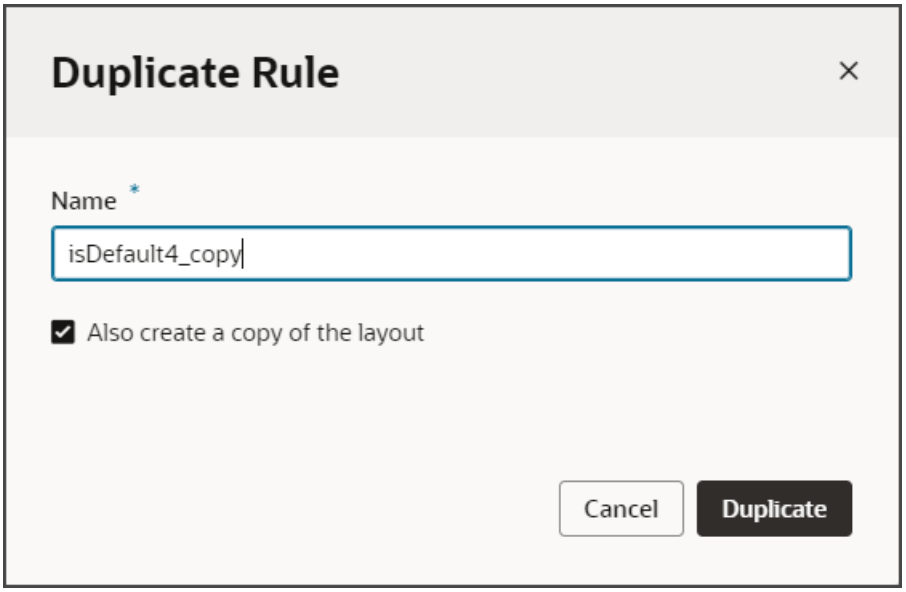
Both a default layout as well as a default rule are displayed for the Sub View Layout.



3. Click the Duplicate Rule icon.

4. In the Duplicate Rule dialog, accept the default rule name or enter a new name. The name you enter here is both the rule name and also the layout name, so enter a layout name that makes sense for you.

   Also, make sure that the **Also create a copy of the layout** check box is selected.



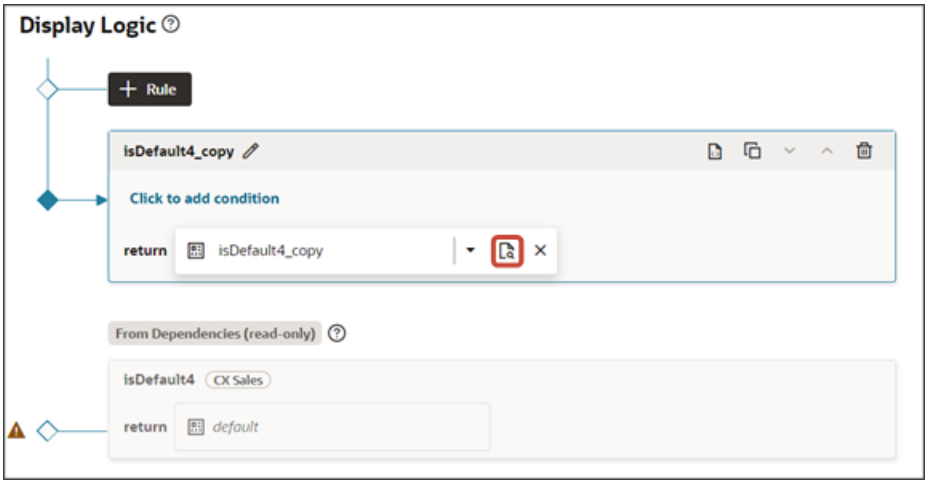
5. Click **Duplicate**.

   The new rule displays at the top of the list of existing rules, which means that this rule will be evaluated first at runtime. If the rule's conditions are met, then the associated layout is displayed to the user.

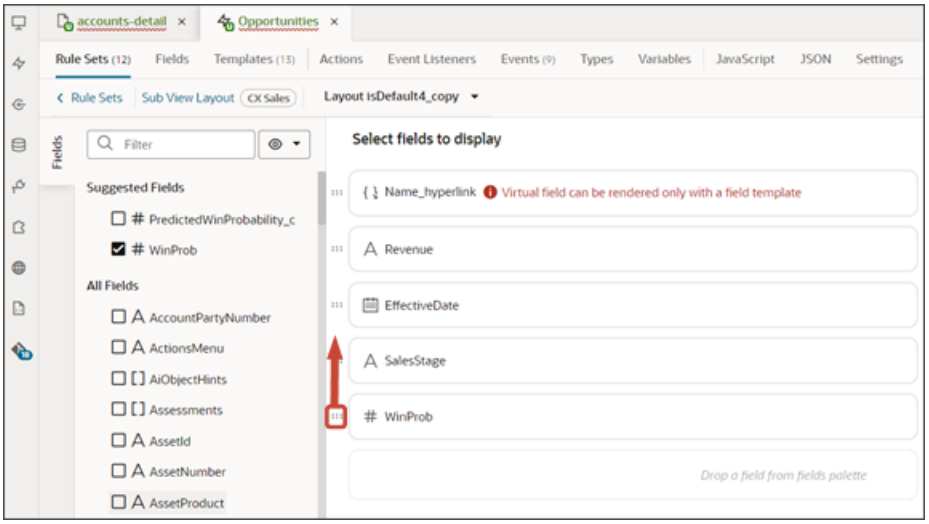
   In this example, we're not adding any conditions which means that the associated layout will always be displayed.

6. Modify the rule's copied layout.

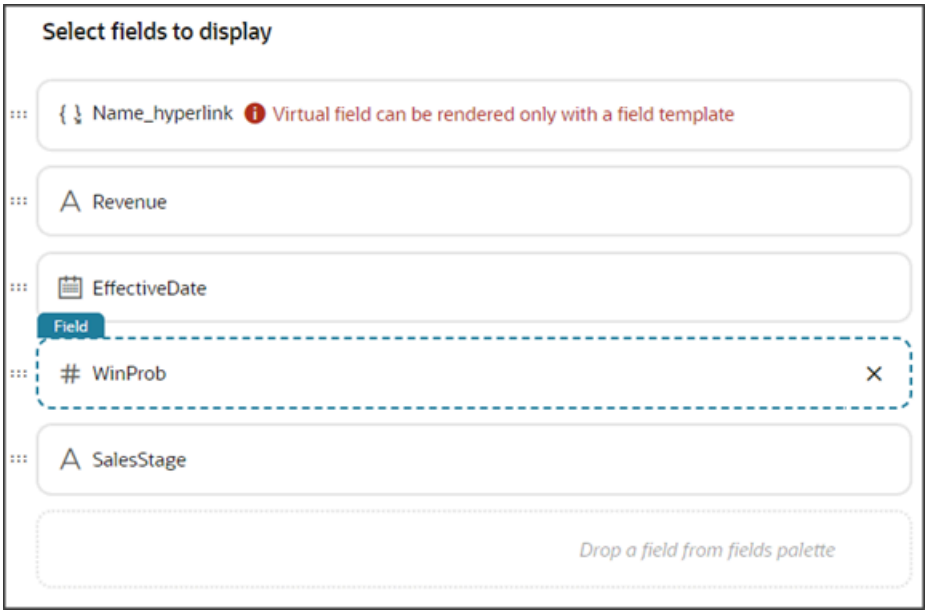
   a. Click the Open icon to edit the copied layout.



   b. In the list of fields in the layout, use the handle next to the win probability field to move it above the sales stage field.



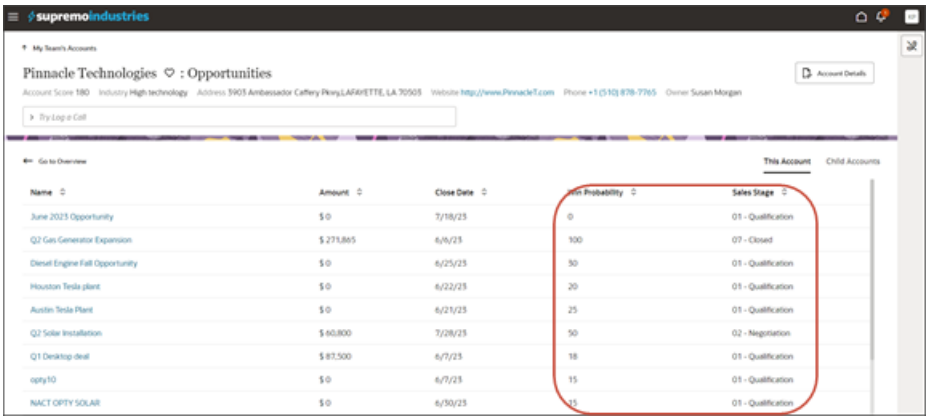
   Here's a screenshot of the final location of the win probability field.



7. Click the Preview button to see your changes in your runtime test environment.

The preview link must include the `application/container` segments in the URL. If not, then change the preview link using the following example URL:

```
https://<servername>/fscmUI/redwood/cx-sales/application/container/accounts/accounts-detail?id=300000003513233&puid=7050&view=foldout
```

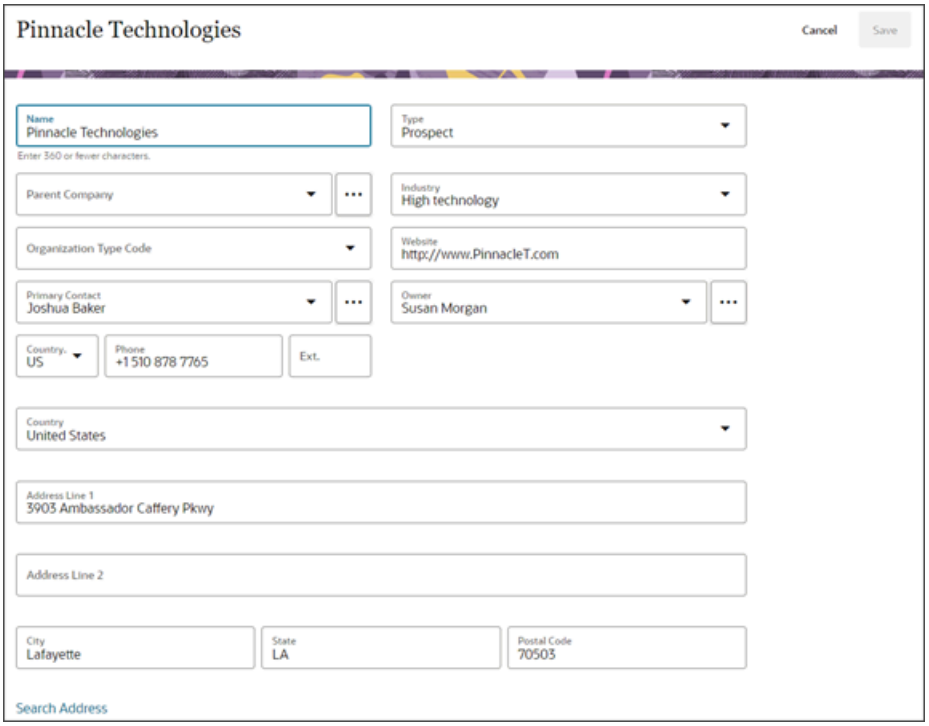


Configure the Create and Edit Layouts

This topic illustrates how to add or remove a field, and how to reorder fields, on the Create and Edit Account pages.

What's the Scenario?

Here's a screenshot of the Edit Account page.



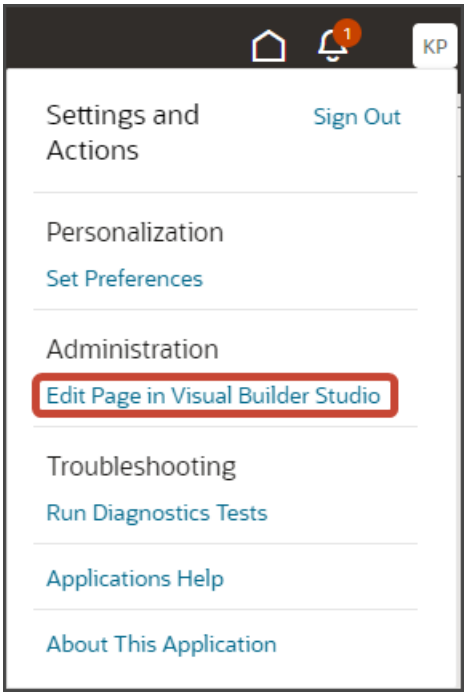
In this example, we'll move the Website field to display to the right of the Name field.

Modify the Edit Account Layout

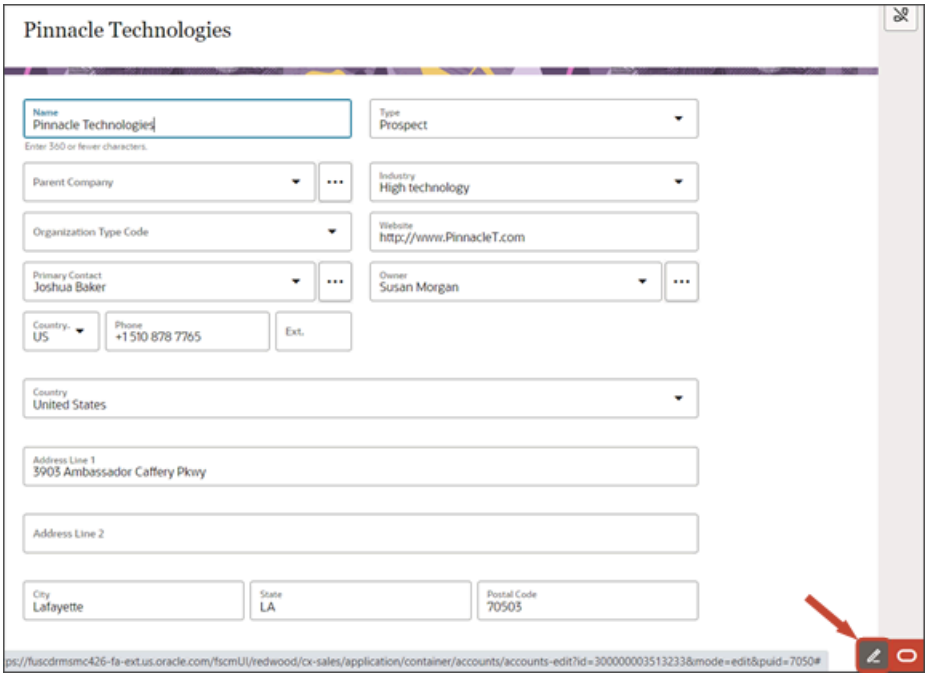
1. Navigate to the page that displays the area you want to extend.

   In this example, navigate to the My Team's Accounts page, click an account record, then click **Account Details**.

2. To access Visual Builder Studio, you have one of two options:
   - Under the Settings and Actions menu, select **Edit Page in Visual Builder Studio**.



   - If you don't see the Settings and Actions menu, then in the lower right corner of the page, hover over the Ask Oracle icon and click **Edit Page in Visual Builder Studio**.

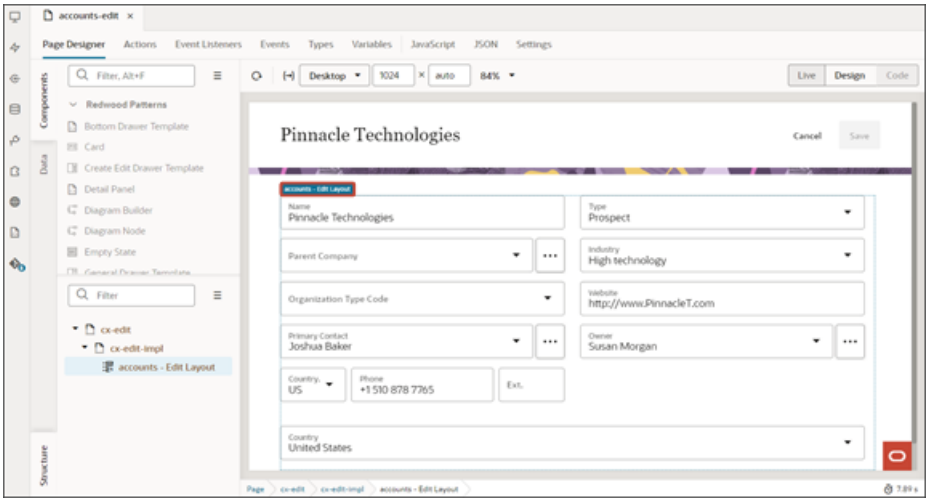


3. If you're working on multiple projects, then select the Extend Redwood CX Sales project to keep your fragments extension separate from non-fragments changes.

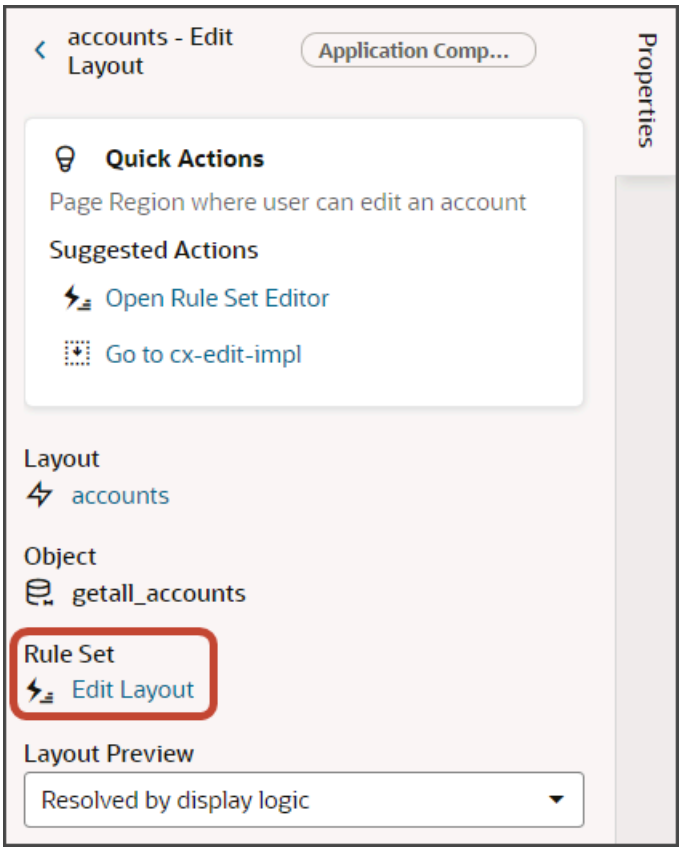
4. When Visual Builder Studio opens, the Edit Account page displays in the canvas.

   On the Structure panel, click the **accounts - Edit Layout** node.

   The edit layout is highlighted in the canvas.

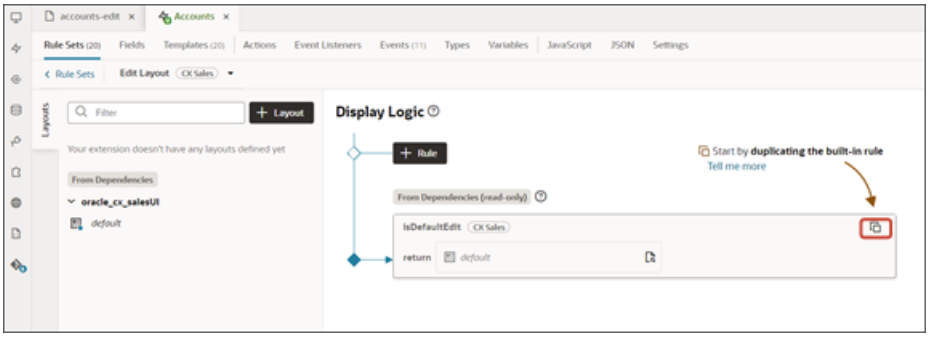


5. On the Properties pane, click the **Edit Layout** rule set.



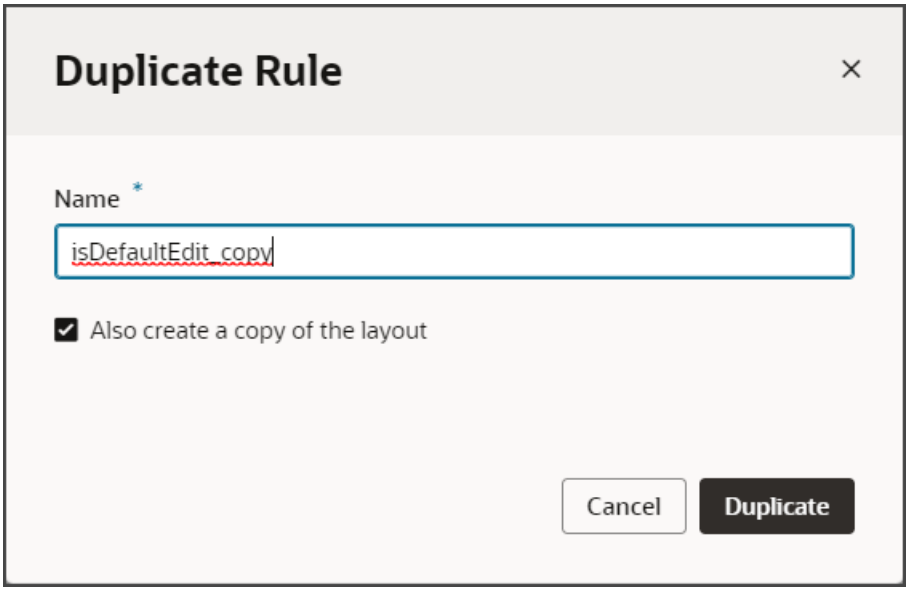
   The Accounts layout tab > Rule Sets subtab displays. The Rule Sets subtab includes both a default layout as well as a default rule. Let's create a new rule.

6. Click the Duplicate Rule icon.



7. In the Duplicate Rule dialog, accept the default rule name or enter a new name. The name you enter here is both the rule name and also the layout name, so enter a layout name that makes sense for you.

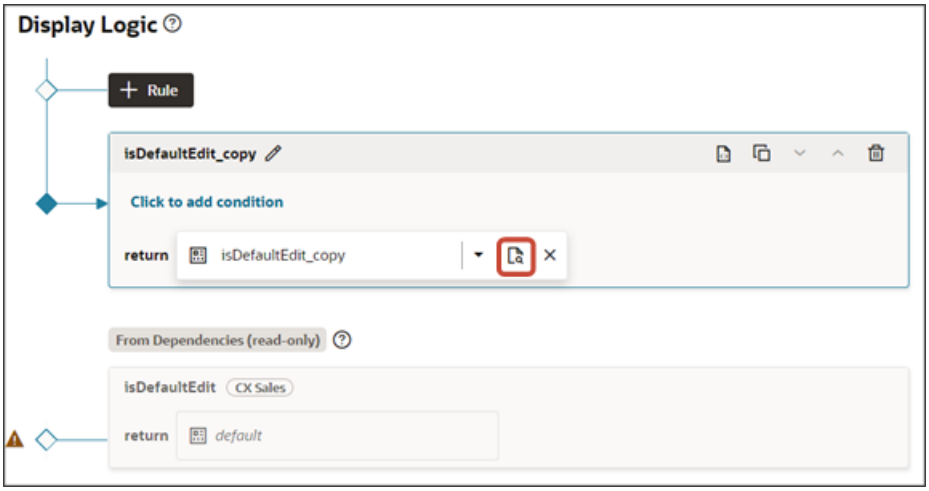
   Make sure that the **Also create a copy of the layout** check box is selected, and then click **Duplicate**.



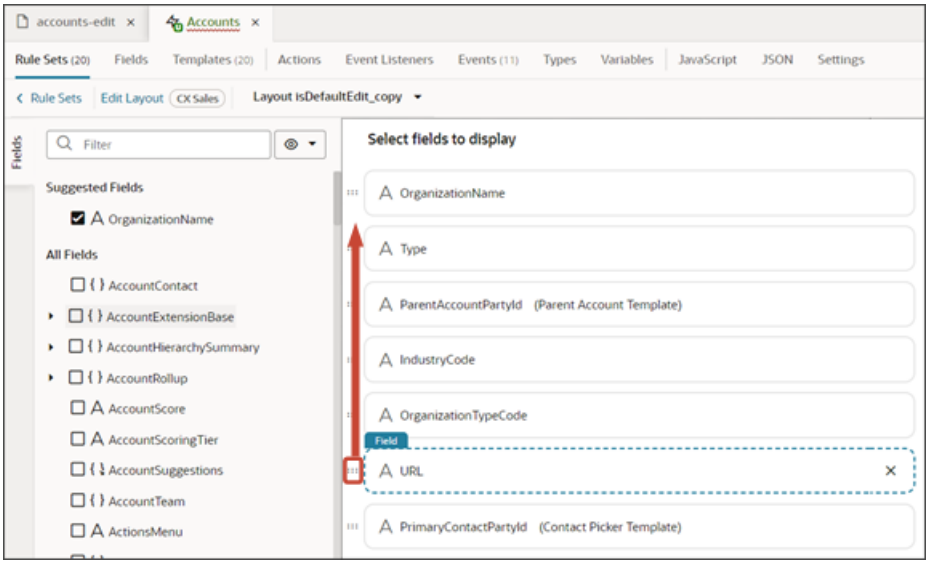
   The new rule displays at the top of the list of existing rules, which means that this rule will be evaluated first at runtime. If the rule's conditions are met, then the associated layout is displayed to the user.

   In this example, we're not adding any conditions which means that the associated layout will always be displayed.

8. Let's modify the rule's copied layout. Click the Open icon to edit the copied layout.



9. In this example, we want to move the position of the Website field. Locate the Website field and use the handle to reposition the field so that it displays after the Name field.

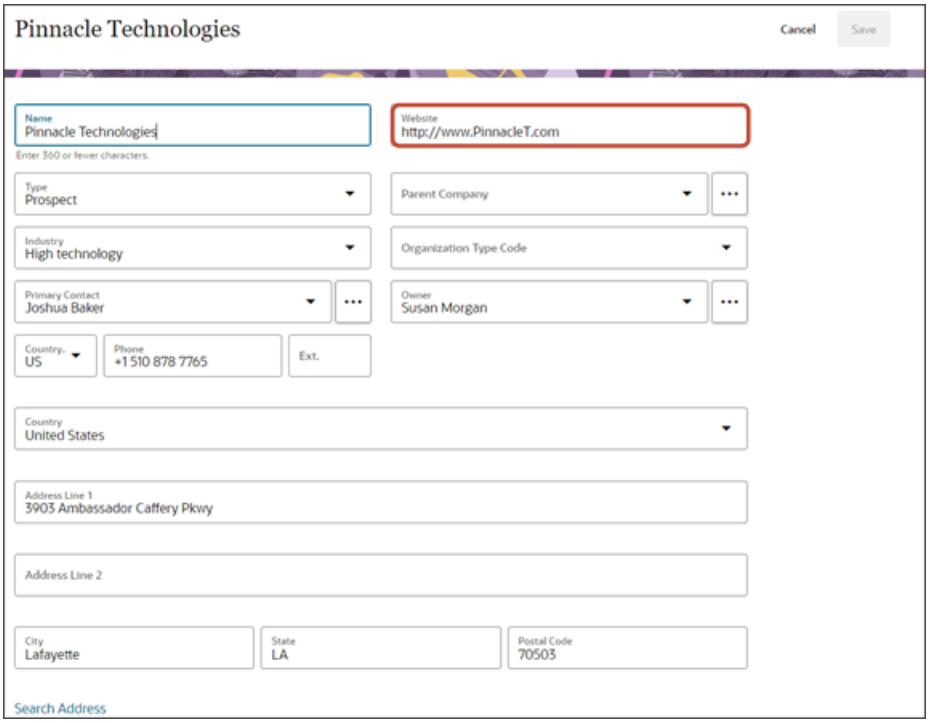


10. Click the Preview button to see your changes in your runtime test environment.

The preview link must include the `application/container` segments in the URL. If not, then change the preview link using the following example URL:

```
https://<servername>/fscmUI/redwood/cx-sales/application/container/accounts/accounts-edit?id=300000003513233&mode=edit&puid=7050
```

This screenshot illustrates the new location of the Website field.



Link to a Smart Action Using a URL

You can construct a URL that calls a smart action in the Redwood version of Sales. Construct this URL whenever needed and then use it as a deeplink. Depending on the smart action added to the URL, clicking the link will either execute a smart action without involving a UI (to delete a record, for example) or navigate directly to an open drawer on a Sales page (to create a record, for example).

To construct the URL, append the smart action ID as a parameter to the detail page URL.

1. Obtain the smart action ID.
   You can retrieve the smart action ID from Application Composer.
2. Obtain the URL of the detail page.
   For example:

```
https://<servername>/fscmUI/redwood/cx-sales/application/container/accounts/accounts-detail?
id=300000008600956&puid=38005&view=foldout
```

3. Append the smart action ID parameter as follows:

```
&actionId=<smart action ID>
```

4. The resulting URL can be used to link to a smart action:

   For example:

```
https://<servername>/fscmUI/redwood/cx-sales/application/container/accounts/accounts-detail?
id=300000008600956&puid=38005&view=foldout&actionId=SDA-Delete-accounts
```

   Note that once the action is completed, the URL changes to:

```
https://<servername>/fscmUI/redwood/cx-sales/application/container/accounts/accounts-detail?
id=300000008600956&puid=38005&view=foldout&actionId=completed
```

Change Navigation to Pages in Your Sales Application

Using the Dispatcher feature in Application Composer, you can change which page opens when a salesperson clicks on a record name link on pages in both standard and custom objects. You can redirect links on the list pages, detail pages, and the edit/create pages. The redirected link can open standard or custom pages and subviews. You can specify different destinations for different job roles.

Clicking the opportunity name link on the opportunity list page, for example, normally opens the opportunity detail page, which provides an overview of key activities, contacts, products, and other information. Getting to what a customer is interested in purchasing requires an extra click. If salespeople are more interested in what the customer is buying than in a general overview, then you can open the subview that lists the opportunity products and revenue directly, saving that extra click.

If you created a simple custom object, you can even skip the detail page altogether and open the edit page instead.

How Dispatcher Works

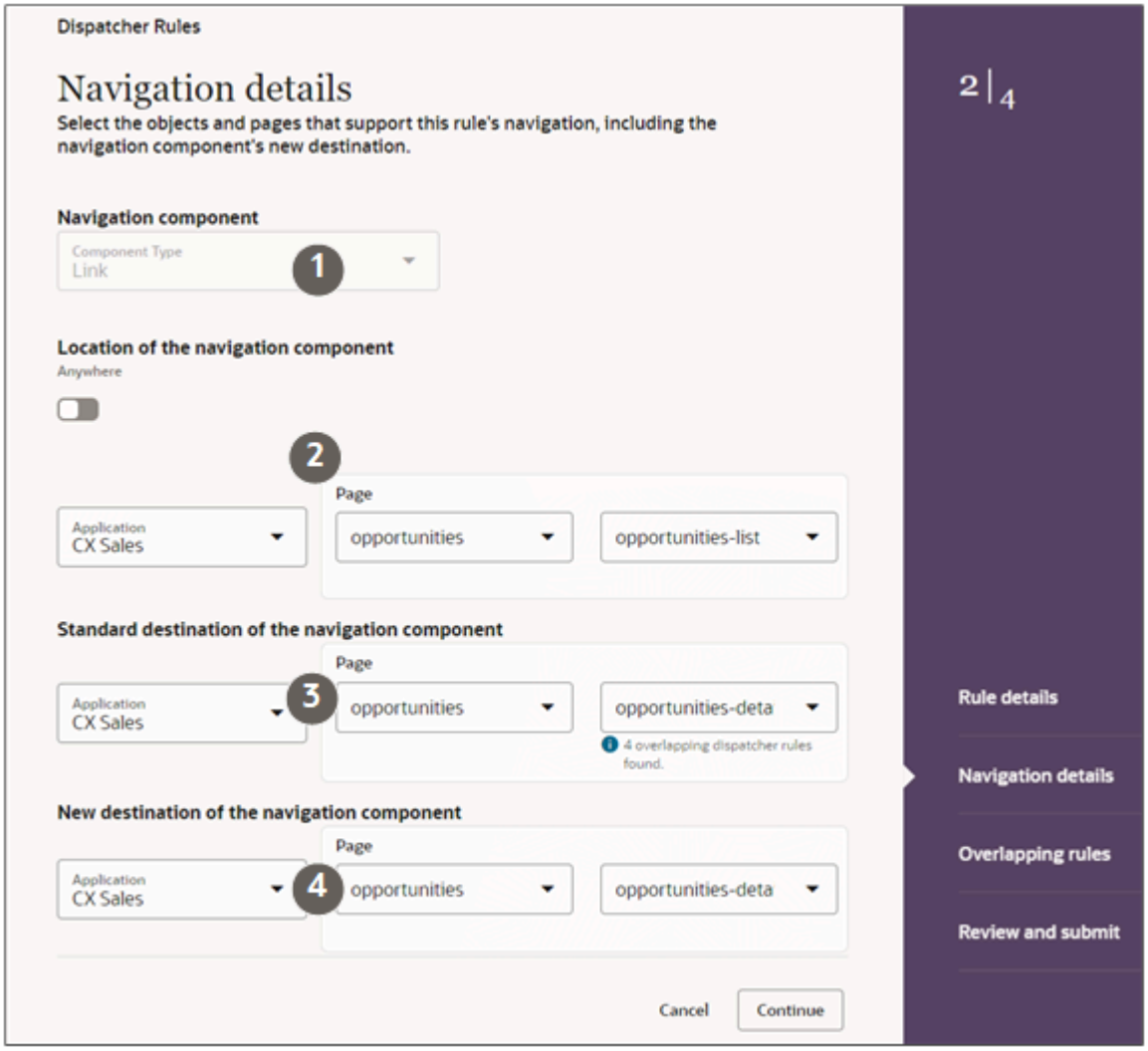
Using the Dispatcher, you can create a set of rules that can open different pages for different job roles. Each dispatcher rule replaces the URLs pointing to the same location. Dispatcher doesn't identify individual links on the page. If a page includes multiple links that go to the same destination, all are replaced. You can even redirect a URL in all the pages in the application to a new destination with one rule.

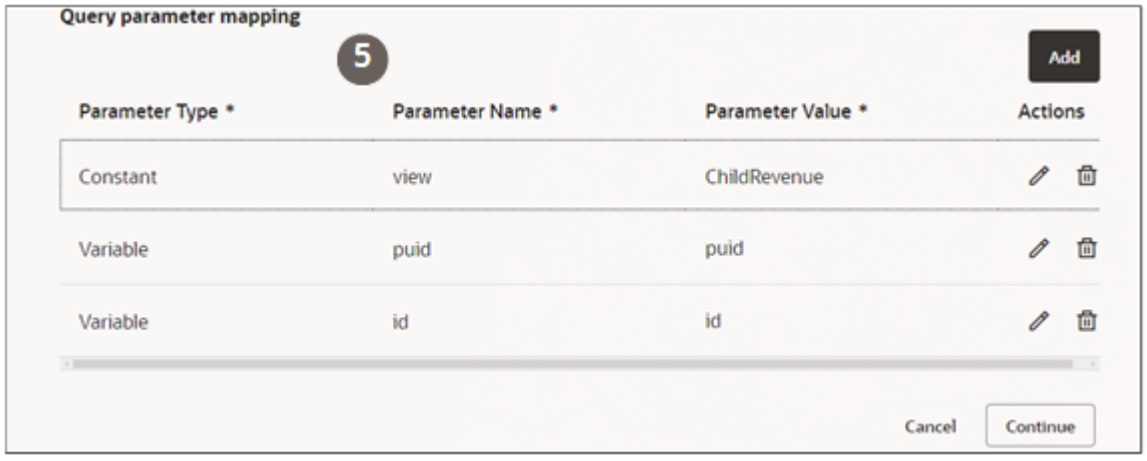
Creating a rule involves 4 steps:

1. Rule Details, where you specify if the rule applies to everyone in the organization or to specific job roles.
2. Navigation Details, where you enter the scope of the redirection rule and both the old and the new destination.
3. Overlapping Rules, where you specify the order in which to process any overlapping rules.
4. Review and submit.

What you enter in the Navigation Details step is key, so here's an overview of the 5 sections in this step. You must scroll down to see the last section. Detailed instructions for creating rules follow.

Section	Description
Navigation Component (1)	In this release, you can redirect only links from the object name link.
Location of the Navigation Component (2)	The scope of the links you want to redirect. You can redirect the links in all the pages of the Sales application, in a specific object, or narrow the scope to a specific page.
Standard Destination of the Navigation Component (3)	The current destination for the link you're redirecting. You can redirect the links on the list page, the detail page, the edit page, and the create page. **Note:** Using Dispatcher, you can't redirect links in subviews.
New Destination of the Navigation Component (4)	The new destination page for the link. Subviews are part of the detail page. So, if you're redirecting the link to a subview, you select the detail page.
Query Parameter Mapping (5)	If you're redirecting a link to a subview, then you identify the subview by adding a constant with a value that you obtain from the subview URL. If you're redirecting to an edit page, you add the constant: **mode = edit**. The variables are standard for all standard objects and custom objects created by the CX Extension Generator.


Dispatcher Rules
Navigation details
Select the objects and pages that support this rule's navigation, including the navigation component's new destination.
2 | 4
Navigation component
Component Type
Link
1
Location of the navigation component
Anywhere
2
Page
Application
CX Sales
opportunities
opportunities-list
Standard destination of the navigation component
Page
Application
CX Sales
3
opportunities
opportunities-deta
4 overlapping dispatcher rules found.
New destination of the navigation component
Page
Application
CX Sales
4
opportunities
opportunities-deta
Cancel
Continue
Rule details
Navigation details
Overlapping rules
Review and submit


Query parameter mapping
5
Add
Parameter Type *
Parameter Name *
Parameter Value *
Actions
Constant
view
ChildRevenue
Variable
puid
puid
Variable
id
id
Cancel
Continue

Example Entries for Redirecting Opportunity List Page Links to the Product Revenue Page

Here's what to enter in the Navigation Details step sections to redirect the opportunity name links on the opportunity list page to the Products subview.

- **Location of Navigation Component**

  You're restricting the redirection to the links on the opportunity List page, so make these entries:

Field	Entry
Application	CX Sales
Page	opportunities
2nd Page field	opportunities-list

- **Standard Destination of the Navigation Component**

  Normally, the application opens the detail page when users click the opportunity name on the List page.

Field	Entry
Application	CX Sales
Page	opportunities
2nd Page field	opportunities-detail

- **New Destination of the Navigation Component**

  You're redirecting the navigation to a subview of the detail page, so your entries are the same as for the standard destination. Subviews are part of the detail page.

Field	Entry
Application	CX Sales
Page	opportunities
2nd Page field	opportunities-detail

- **Query Parameter Mapping**

  To redirect to the Product subview, you add a constant with the value of ChildRevenue:

Field	Entry
Parameter Type	Constant
Parameter Name	view
Parameter Value	ChildRevenue

Steps to Create and Activate Dispatcher Rules

1. Open Application Composer outside a sandbox.
2. Click **Dispatcher**.
3. On the Dispatcher page, click **Create**.
4. In the Rule Details page, enter a name for the rule.
5. In the **Rule Conditions** section, specify the audience for the rule. You have two options:
   - Make the rule apply to the all job roles in the organization by turning on **Apply Rule Globally**.
   - Apply the rule to specific job roles you enter in the **Role Filter** field.
6. Click **Continue** to move to the **Navigation Details** step.
7. In the **Location of the Navigation Component** section, specify the scope of the rule:
   - To have the link redirected on all pages, turn on **Anywhere**.
   - Narrow the scope of the redirection to an object and page:
     - In the **Application** field, select either **CX Sales** for standard pages, or **CX Custom**.
     - In the **Page** fields, make these selections:
       a. In the first **Page** field, select the object.
       b. In the 2nd Page field, specify the page type:

Available Values	Description
any	Redirects links on all pages for the object.
list	Redirects links on the list page.
edit	Redirects links on the edit and create pages.
detail	Redirects links on the detail page.

8. In the **Standard Destination of the Navigation Component** section, enter the current navigation destination. Your entries identify the URL to be replaced.

    a. In the first **Page** field, select the object.
    b. In the 2nd Page field, select the page.

Available Values	Description
detail	The detail page (called the Overview page at runtime).
edit	The edit/view page.
list	The list page.

9. In the **New Destination of the Navigation Component** section, enter the new navigation destination.

    a. In the first **Page** field, select the object.
    b. In the 2nd Page field, select the page.

Available Values	Description
detail	Redirects to the detail page or subview.
edit	Redirects to the edit or the create page. If you're redirecting to the edit page, then you must also add the constant **mode = edit** in the Query Parameter Mapping section. If you don't add a constant, the user is redirected to the Create page.
list	Select to redirect to the list page.

10. If you're redirecting the link to a subview or to the edit page, then you must add a constant in the **Query Parameter Mapping** section:

    a. Click **Add**.
    b. If you're redirecting to the edit page, then make the following entries:

Field	Entry
Parameter Type	Constant
Parameter Name	mode

Field	Entry
Parameter Value	edit

c. If you're redirecting to a subview, then enter the following:

Field	Entry
Parameter Type	Constant
Parameter Name	view
Parameter Value	Enter the last part of the subview URL following **`view=`**.

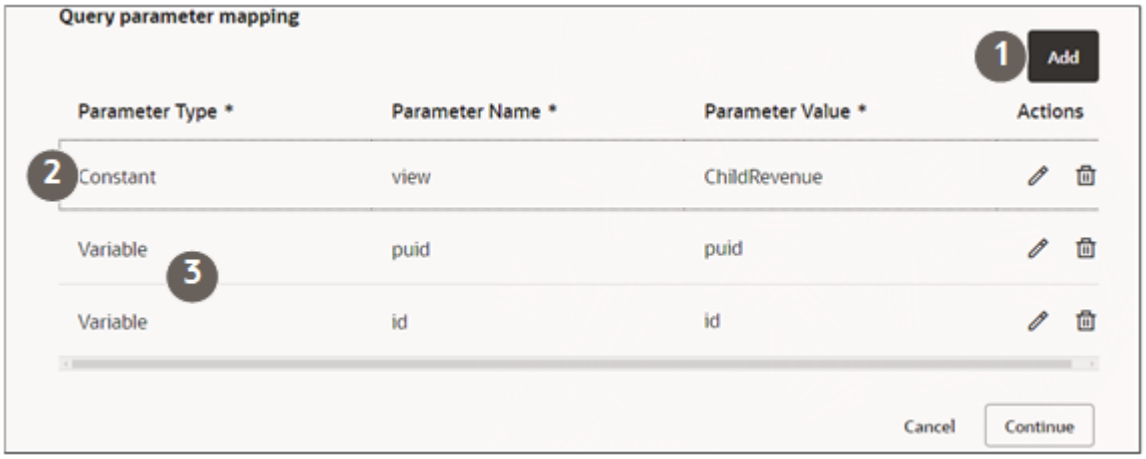
Here's an example of a URL for the Products subview on an opportunity:

```
https://<domain>/fscmUI/redwood/cx-sales/application/container/opportunities/opportunities-detail?id=300000009863286&puid=39003&view=ChildRevenue
```

> **Note:** For standard subviews and subviews generated by the CX Extension Generator, the application automatically adds 2 parameters: the variables **puid** and **id**. These parameters are required.

Here's a screenshot of the Query Parameter Mapping section

Callout	Description
1	The **Add** button.
2	Constant entry.
3	The 2 required variables are included automatically.



11. Click **Continue** to move to the **Overlapping Rules** step.
12. Review the order of any rules with overlapping functionality and specify the order of priority by dragging them into position using the handles on each row. The rule at the top gets executed first.

13. Click **Continue** to move to the **Review and Submit** step.
14. Click **Submit**.
15. On the Dispatcher list page, select **Action** > **Mark Active**.